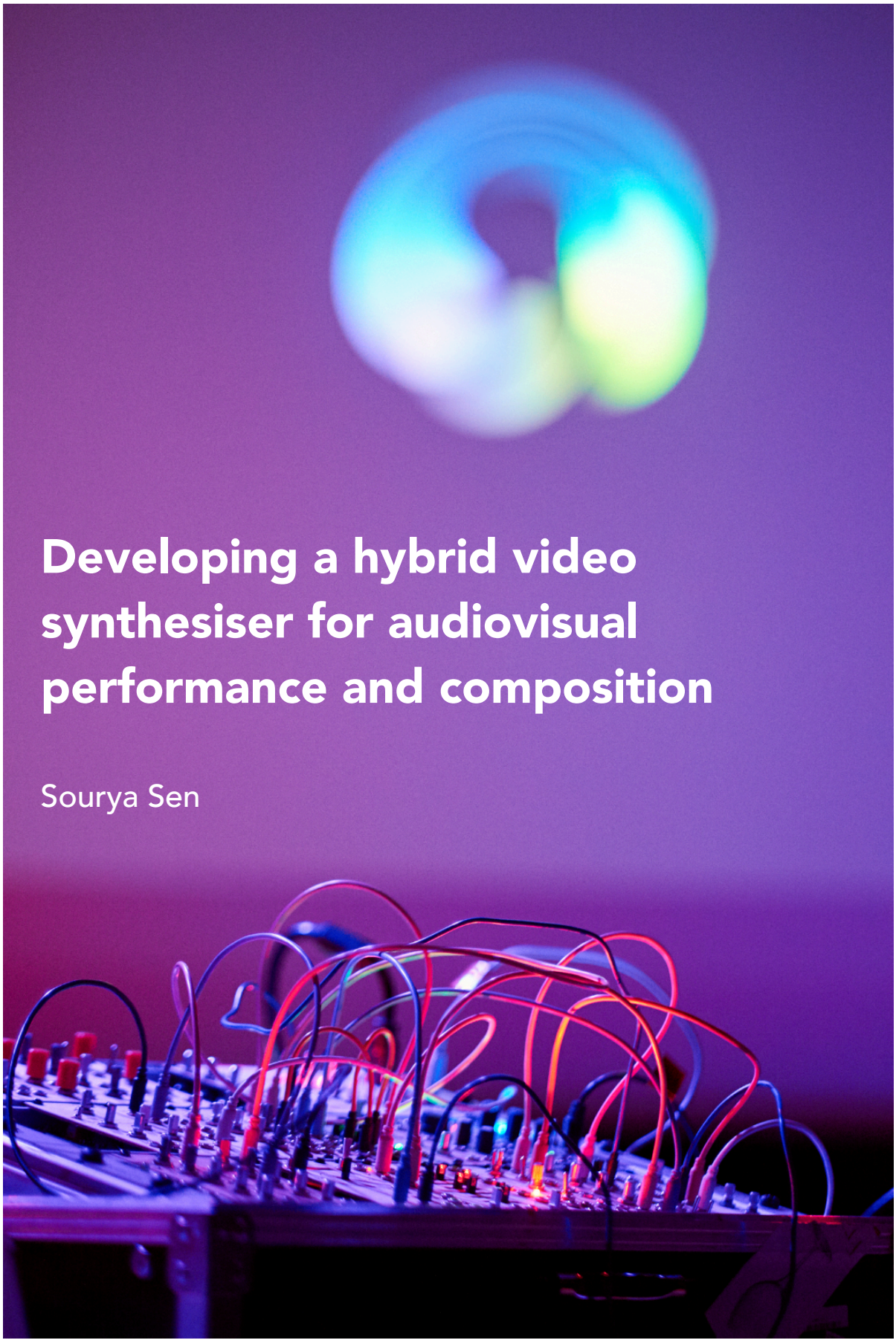


Developing a hybrid video synthesiser for audiovisual performance and composition

Sourya Sen

Developing a hybrid video synthesiser for audiovisual performance and composition

Author: Sourya Sen

Thesis advisor: Nuno N. Correia

Thesis supervisor: Koray Tahiroğlu

Submission: April 2019, Espoo, Finland

Programme: Masters in New Media Design and Production, Aalto University

TABLE OF CONTENTS

ABSTRACT

This thesis presents the development process and analysis of a hybrid video synthesiser that was designed towards use in audiovisual performance and composition.

Though there are various instruments, platforms, and approaches towards audiovisual performance and composition, this practice-based thesis, through the development of an instrument, addresses a very specific gap – the lack of a hybrid, i.e., a software-based physical modular video synthesiser.

This thesis will trace the historical origins of the video synthesiser, contextualise audiovisual performance and composition, and discuss the development of this hybrid video synthesiser. This thesis will document the iterative development process, provide key takeaways from performances and finally, provide a discussion, certain recommendations, and contributions that this instrument makes within the field of video synthesisers and audiovisual performance.

Keywords: audiovisual, audiovisual performance, video synthesis, video synthesiser, openframeworks, raspberry pi, hardware, modular, live, performance, composition, open source, new media, software, framework, Eurorack

1. INTRODUCTION

Live audiovisual performance can be defined as an event of sound and image manipulation (Carvalho, 2015, p. 162). In the last few decades, mainly through technological advancements, it has gained prominence, both in entertainment and in research (Carvalho & Lund, 2015, p. 7). Various strategies, platforms, and instruments exist towards composing and performing audiovisual material (see chapters 2 and 4). As the field of new media combines media technologies and digital computing (Manovich, 2018), and audiovisual performance has ties to the available technology (Carvalho & Lund, 2015), it serves as a platform for inquiry towards the development of such instruments and platforms. Even with the advancements in computing technology and the development of new instruments that exist, however, there are still gaps that need to be addressed towards certain applications (see chapters 1 and 4). This thesis, through the development of a hybrid video synthesiser, tries to address such a gap as observed in my personal practice in the field of audiovisual composition and performance. This chapter provides the motivations, preceding work, and defines the aims of the project to the reader.

1.1 MOTIVATION

The motivations for this thesis come from personal interest. Trained in audiovisual production during my undergraduate studies and having had an interest in music since childhood (including having played in a few amateur bands), before coming to Aalto University I was working actively in the Indian indie music sector. This included working on video post-production for various music related projects like music videos, music documentaries, etc., and through it, working and performing with various music acts as a visual jockey (VJ) under the alias Oblique. Among these things, I also worked with several indie music festivals in designing stages and visuals for their properties (Superdry Krunk Live Sessions on MTV Indies, Magnetic Fields Festival, 2014 and 2015, etc).

Figure 1: Live visuals performance setup, 2015-16

Through my practice as a VJ and generally working closely with artists and producers as a visual artist and also working in video post-production, the dichotomy that existed in both these audiovisual forms became evident. Working on music videos or as a VJ, the video served as a complementary function to something that already existed, the music. On the other hand, film and video production relied on music directors to compose audio for visual sequences once the video already existed in some form, this time the music serving a complementary function.

While these established practices serve their functions well, personally being invested in the world of video and having an interest in music production led me to explore my own audiovisual practice. Instead of composing one for the other, I wanted to explore composing both simultaneously. The first step I took in this regard was my 2015 audiovisual work, *Berlin: Not at War* (Sen/Oblique, 2015) which was a result of composing audiovisual loop-based material simultaneously and having them closely relate to each other.

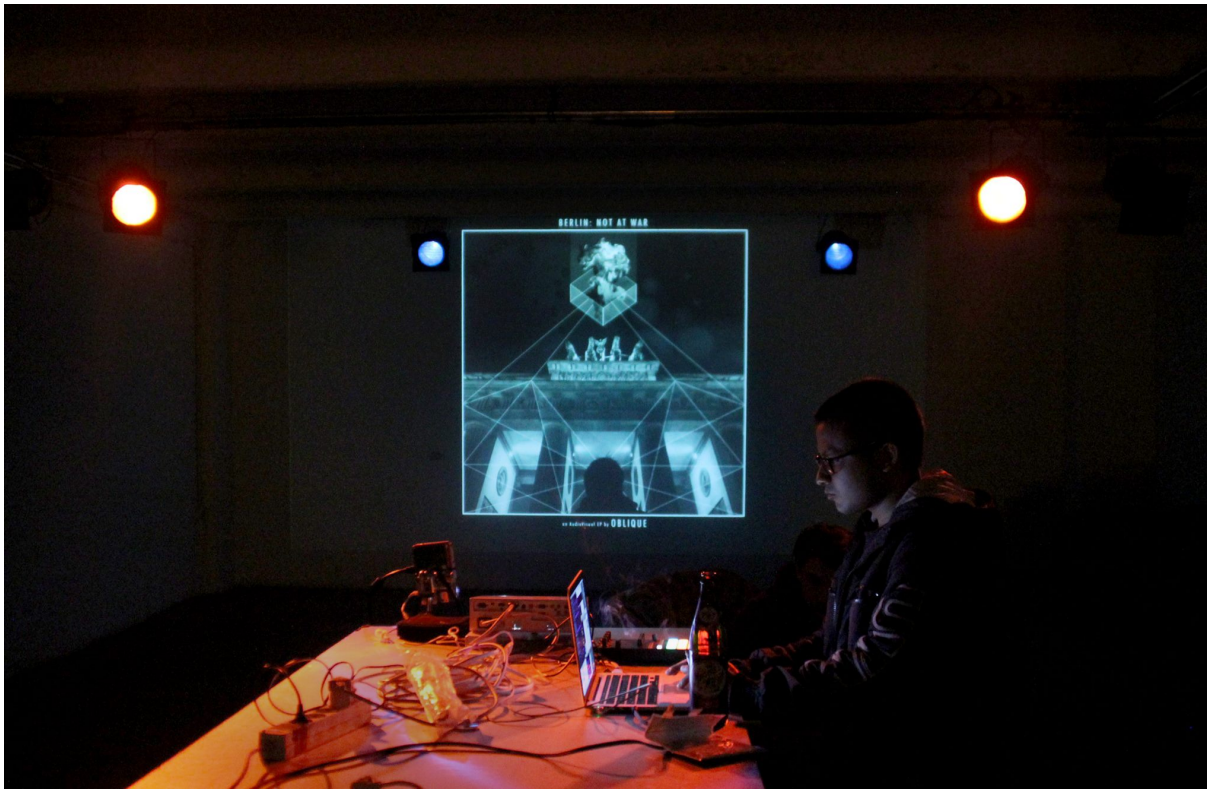


Figure 2: Presenting Berlin: Not at War, VJ Union Meeting, Berlin 2015

My interest in composing electronic music also started with this project, as I had to compose music and work with video at the same time. I worked on music production using Ableton Live, a popular digital audio workstation (DAW). Even though I continued working on the DAW after this project, I never fully identified with a computer-based approach towards producing music. I was always missing the immediacy of a musical instrument and this process always felt unnatural. A few hardware MIDI controllers did help, but the practically unlimited options with virtual studio technology (VST) instruments and presets, as well as the grid-based workflow and other features unique to the DAW, never felt inspiring to me.

Simultaneously, with my visuals practice I was facing certain limitations. As a VJ, I was preparing clips and loops beforehand for a performance, and while I was performing them live, the inability to truly generate visuals was becoming restrictive. I wanted to leave the world of loop-based video performances behind and explore real-time composing of visual material.

Since coming to Aalto University, these two have been my main focus of inquiry through different techniques and approaches. Namely, a music practice that is away from the computer and developing a system for visual performances that rely on real-time generative material that are not entirely composed beforehand. This thesis is the culmination of these two goals through practice-based research.

1.2 TOWARDS AN ORGANIC AUDIOVISUAL PRACTICE

My studies at the Department of Media towards a major in New Media Design and Production has played a huge role towards this particular thesis project. Specific courses as well as projects by my peers, colleagues and the research groups within the department have directly and indirectly influenced me as well. As part of my studies, I have gained an understanding in designing both digital and physical tools, from programming to electronics, combining virtual and physical worlds, and working with digital fabrication and interaction. More specifically, working with microcontrollers and projects like the ofxPiMapper (Rijnieks, 2015), various approaches towards sound and music interaction (Aldunate Infante, 2018; "Sound and Physical Interaction | SOPI Research Group," n.d.) and audiovisual projects like Vector Synthesis (Holzer, 2017), have had a role to play towards the continued development of my practice. Specific courses like Programming for Artists, Electronics for Artists, Generative Media Coding and Computer Graphics, etc., have provided opportunities to explore various strategies towards audiovisual art and have shaped my skills leading to this project.

Through the span of these courses, in more practical terms, towards developing a musical practice that felt more organic, I have been working with DIY electronic circuits, developing my own musical instruments, as used in my 2017 series of performances, *Pattern Noise*, performed at LPM 2017 and Generate 2017, among other places.

Figure 3: Pattern Noise performance, 2017

These circuits, primarily based on CMOS chips, while able to produce sounds, were not often musically controllable and presented difficulties in creating pitched melodies and compositions. To achieve such goals on a physical musical platform, I started working with the Eurorack format, first building my own modules and later purchasing commercial modules available for that system. The biggest appeal of this modular and physical format was having hands-on control over every aspect of the outcome and also a thorough understanding of every piece in the system that made up the composition or performance environment.

Additionally, I was exploring generative art and visuals through creative coding practices. These can be seen in personal daily studies over a period of a few months on my Instagram profile, as well as developing tools and libraries towards the same. One of the bigger projects done through these means was the branding elements of Aalto Day One (Sen, 2018).

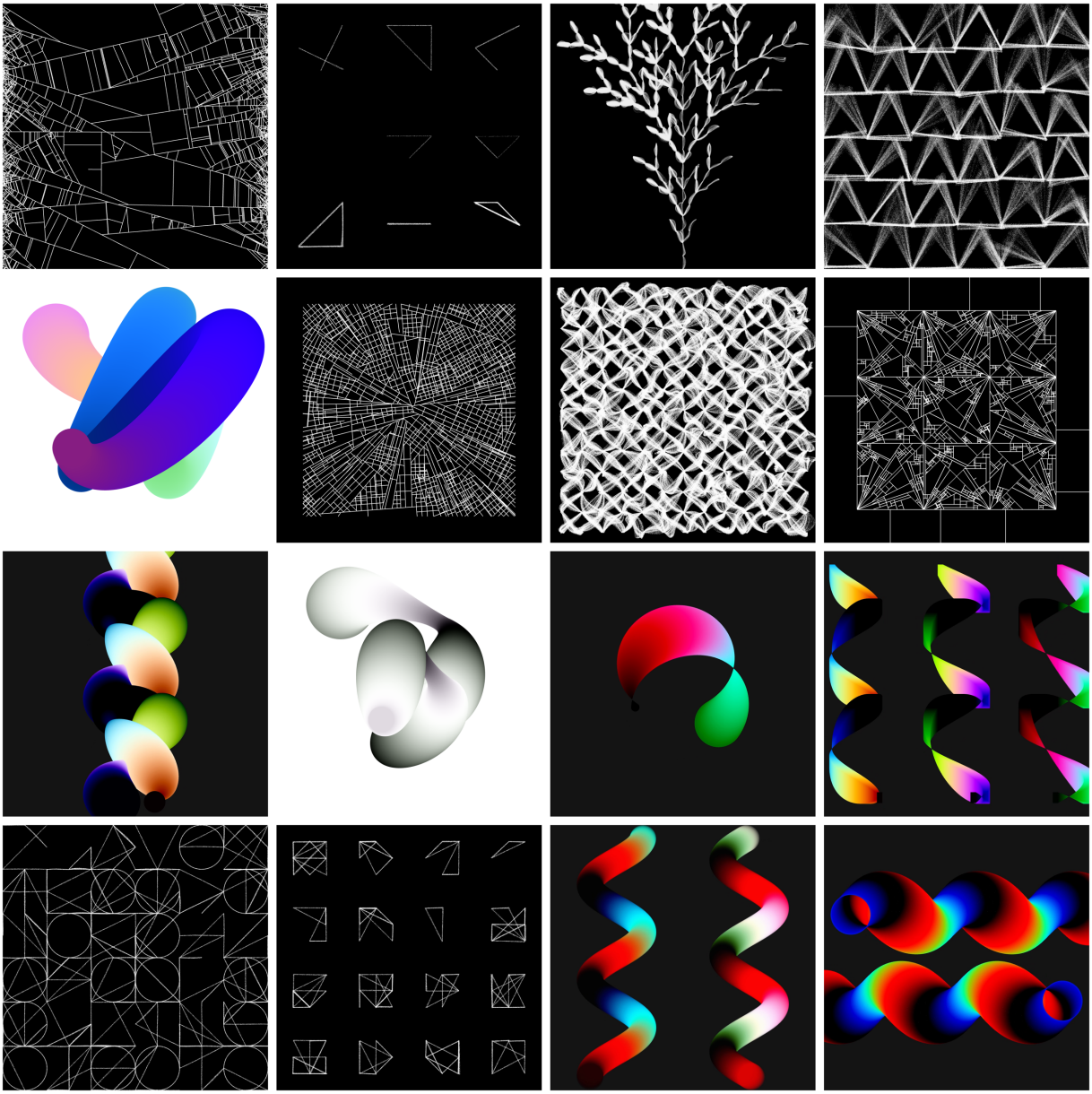


Figure 4: Examples from generative art practice, 2017-18

This thesis is an attempt to bridge these two mediums and develop a hybrid instrument, i.e. combining the physical and digital worlds I was working in, towards the application of audiovisual performance and composition practices.

1.3 THESIS AIMS AND OBJECTIVES

The primary aim of this thesis project was to develop a video synthesiser that bridged the gap between hardware visual synthesis tools and contemporary digital practices towards audiovisual performance. More specifically, while my personal music practice was based on modular hardware synthesisers, and my visual practices were based on digital tools and platforms, I wanted to combine the two methods towards a hybrid audiovisual performance practice through the development of this video synthesiser.

With regards to additional objectives, I wanted this instrument to be a platform that was both a plug-and-play solution and also offered a wide range of customisation. I did not want to just build an instrument for myself but make it easily approachable for everyone which is why it was also important to develop this open source and keep it economically viable. Often, new instruments and platforms develop their own ecosystems and paradigms, which make it difficult to be immediately accessible or need a certain amount of time to develop as a standard, which is why I did not also want to work in a vacuum but develop it within the Eurorack standard, such that there were already an existing network of tools, modules and acceptable standards to take advantage of.

1.4 SCOPE OF THE THESIS

Defining a scope for a project like this is difficult, as it can go in many directions. Apart from discussing the process of design and development, it can look into various aspects like interactions in a modular environment, instrument design, electronics, software development, analysis of existing systems, computer graphics, analogue video, and more. As it would be impossible to investigate every route, careful consideration has been taken to write about the processes and influences that are directly relevant and only touches upon aspects that have a major role to play in the design and development of this instrument. Furthermore, the topics covered in this thesis do not aim to or claim to be an exhaustive study into the same. What this thesis aims to do, however, is to provide the background for the necessity of this instrument, looking at historical landmarks and their influences, which have further influenced this project

and provide a framework for analysing the major developments that this project accomplishes, while discussing the design and development process. Furthermore, certain discussions, observations and recommendations are offered, and finally, conclusions are drawn up towards the end, providing critical analysis into the subject matter.

1.5 STRUCTURE OF THE THESIS

This section discusses the structure of the thesis and is part of the introduction or chapter 1. This chapter briefly introduces the field, my motivations and the overall aim of the thesis project.

Chapter 2 discusses the relationship between audio and video, and the paradigm of audiovisual composition and performance – the context within which my instrument is meant to be used. Furthermore, chapter 2 traces the evolution of the video synthesiser as an instrument, including defining the term and tracing the nature of its inheritance from analogue modular music synthesisers and the role played by the digital nature of computer graphics. The contemporary landscape is also briefly discussed here and the nature of the current offerings towards video synthesis and music composition briefly touched upon (a more detailed analysis is presented later in the thesis).

Chapter 3 discusses the methodology that was followed towards the development of the instrument, the timeline and the role of this thesis. Based on the methodology, chapter 4 analyses the contemporary landscape of audiovisual performance and the strategies used towards the same. The characteristics of audiovisual instruments are identified and granular design requirements are drawn up towards fulfilling the aim of the thesis.

Chapter 5 discusses the development process of the instrument, broken down into hardware and software development through two distinct periods. Chapter 6 discusses the various performances in which this instrument was used and presents the key takeaways from each performance that led to further refinements in the development process. These two chapters

also provide a summary of the development process and the state of the instrument as well as some patching strategies implemented towards performing with this instrument.

Finally, chapter 7 serves as a textual analysis, self-evaluating the instrument towards the requirements set, provides some observations and recommendations and also discusses the contributions made towards the field by releasing the project open source. Chapter 8 wraps up the thesis, discussing future plans and provides some final thoughts with regards to the development process of the instrument.

2. VIDEO SYNTHESIS AND AUDIOVISUAL PERFORMANCE

This chapter discusses the practice of audiovisual composition and performance and the historical development of video synthesisers. While the two may seem interrelated, they can also exist separately, and the distinctions and the relationships that are relevant to this thesis are discussed here.

2.1 AUDIOVISUAL RELATIONSHIPS

Searching for correlations between audio frequencies and colour hues has a long history (Moritz, 1997). The earliest discussions can be found in the writings of ancient Greek philosophers like Pythagoras and Aristotle (Ciufo, 2002). The earliest machines that were invented to explore these relationships were built in the 1700s (ibid.), and the visual instruments themselves were mechanical constructions and optics that would "display modulated colored light in some kind of fluid fashion comparable to music" (Moritz, 1997).Through this, it is clear that a goal to perform visuals, like performing music, is not a new concept and was only limited by the technology of their time. These are precursors to what today has developed into its own field of research and performance.

The earliest machines, termed "colour organs" (Moritz, 1997), continued to be built during the 20th century and used in concerts and performances (Ciufo, 2002). While Ciufu points out that the performances using this sort of mechanical instruments were not well received, he also notes investigations in sound-image relationships continued through other manifestations of colour organs and also other means like film and graphic scores.

Referred to as visual music today (Carvalho & Lund, 2015), these earlier forms of audiovisual experiments have led to various disciplines, namely, expanded cinema, live cinema, VJing and live audiovisual performances (ibid., p. 5). While the boundaries within these practices are blurry, a distinction and definition is still valuable for the purposes of this thesis. Carvalho and Lund (2015) write:

> Generally speaking, audiovisual works range across media such as TV, cinema and live shows to include all the possibilities that present a stimulus to both auditory and visual sensorium systems. … To refer to a work as audiovisual already implies an intermedia connection, one whose very nature lies in the combination of the two words put together: audio and visual. By itself, 'audiovisuality' is not an artistic practice but describes a generic group of practices. (ibid., p. 11)

Therefore, any intermedia work comprising of audio and visual material can be termed as audiovisual. When it comes to artistic practices arising from the same, however, further distinctions can be made.

The Pythagorean concept of arranging colours in intervals similar to a system of organizing musical frequencies towards a practice of visual music as well as incorporating concepts of synaesthesia are often attributed to the development of an audiovisual practice (Carvalho & Lund, 2015; Moritz, 1997). However, those concepts are not adequate towards defining the field. In reality, contemporary audiovisual production is an amalgamation of achievements in technology (Carvalho & Lund, 2015; Ribas, 2011). Furthermore, Carvalho and Lund (2015) define live audiovisual performance as a "term applied to contemporary artistic expressions of live manipulated sound and image, defined as time-based, media-based, and performative. Live audiovisual performance is complex because it does not comprise a specific style, technique, or medium, but instead gathers a series of common elements that simultaneously identify a group of artistic expressions as well as specific works, which don't necessarily fit within either of the particular expressions that constitute the group." (p. 124) This definition is rather broad and takes into account various methods and means that may constitute an audiovisual practice. Chion (1994) makes the argument towards combining audio and visual elements to create a third "audiovisual element", which creates "added value". Elaborating on the same, Grierson (2005) defines an audiovisual practice as the "process of composing … which exploit added value" combining audio and visual material and "is a study which focuses precisely on the nature of

these combinations" and in this case, the individual aural and visual elements themselves only exist to fulfil the other and do not serve a purpose independently (Grierson, 2005).

Grierson (2005) also discusses the various strategies that are often used to compose audiovisual material. According to him, these can be of three types, namely, synchronisation of audio and visual elements, audiovisual congruence, and, binary opposition. His discussion on this topic (2005, p. 24) concludes with citing and agreeing with the model proposed by Cook (1998) regarding combination of audio and visual material. To paraphrase, they either relate to each other, oppose each other or are aligned with each other. These relationships between audio and visual material in an audiovisual composition come together to create added value and is a valuable framework for analysis. It is interesting to note, however, that artistic intent is not taken into consideration here. The content of the audio and visual material coming together to create this audiovisual piece is often abstract in nature and contrary to other artforms, the meaning behind the work is not part of the discussion.

2.2 THE ELECTRONIC ANALOGUE VIDEO SYNTHESISER

Audiovisual production and audiovisual performances are linked to the technical advances of their time. During the 1960s, the medium of television gave rise to the field of video art and the electronic video synthesiser (Collopy, 2014). Television, alongside the availability of affordable electronics and the post war abundance of discarded equipment led to many artists experimenting with the manipulation of images on the screen, either by directly influencing the video signal or by interfering with the cathode ray tubes in the television itself (ibid. p. 74-75).

Collopy argues that "designers modelled video synthesisers on audio synthesisers…" (2014, p. 74). The earliest analogue music synthesisers were being built at the same time, and it gave a point of reference for designers to base their video synthesiser designs on. Thus, to understand the evolution of video synthesisers, it is important to understand the evolution of audio synthesisers first.

The term synthesiser is a matter of debate. Dunn (1992) notes, "there is nothing synthetic about the sounds generated from this class of analog electronic instruments, and since they do not 'synthesize' other sounds, the term is more the result of a conceptual confusion emanating from the industrial nonsense about how these instruments 'imitate' traditional acoustic ones." (Dunn, 1992, p. 37) Therefore, a synthesiser is not meant to sound like a traditional instrument like a flute, violin or piano, even though it can. Any electronic device which can generate sound as its primary function can be called a synthesiser. The term, however has survived in the common vocabulary. Extending the concept, the voltage controlled analogue synthesiser is a collection of different circuits or modules with functions ranging from sound generation (oscillators, noise sources) to modifiers and utilities (Dunn, 1992). The circuitry between these modules are not fixed and they can be determined by the performer or composer through the means of patch cords. Control over these circuits are either physical knobs, buttons and other physical interactions or through these patch cords using "control voltage" – a concept developed by the earliest pioneers of synthesiser designers like Robert Moog and Don Buchla and incorporated independently in the Moog Modular Audio Synthesiser (1964) and the Buchla 100 Series (1964) respectively (Trocco & Pinch, 2009). Control voltages are *the* backbone of a modular synthesis platform. Instead of manually turning a knob, using the output voltage of one circuit, or a module, to control another module enables a composer or performer to automate events towards musical results. Furthermore, control voltages are, ultimately, just voltages. This allows any output to connect to any input – a crucial concept when it comes to understanding modular synthesis paradigms. Control voltages are still used today in modular synthesisers (ibid.) and the concept was adopted by the earliest video synthesiser designers as well (Collopy, 2014, p. 74-75).

The term synthesiser, when it comes to video, is similarly misleading. If any electronic circuitry capable of producing sound can be termed an audio synthesiser, any electronic circuitry capable of producing video signals can be called a video synthesiser. Video synthesis therefore is not about synthesising real images, even though they can be, but encompasses any sort of video that can be displayed or projected. The earliest forms of video synthesis involved feedback effects and direct manipulation of the video signals, and were being simultaneously developed and practised in the 1960s (Collopy, 2014). The true video synthesiser, for the purposes of a

definition, would come from Eric Siegel in 1970, with his invention of the Electronic Video Synthesiser. In his own words, it was:

> like the video equivalent of a music synthesizer, where you have a program board and you can start to set up a whole series of visual geometric happenings in color on the video signals—the screen—and this is designed for video composition. . . everything is composed right inside of the synthesizer. (as quoted in Dunn, 1992)

This thesis will use the term video synthesiser with a similar meaning, a machine capable of creating video signals without the need for an external video signal.

Siegel was far from the only designer, artist or inventor working with video synthesisers. Others like Nam Jun Paik, Bill Etra and Steve Rutt, Dan Sandin, Stephen Beck, etc., were also active at the same time developing their own video synthesisers (Collopy, 2004, p. 74-76).

One of the defining factors in these synthesisers is the continued influence from music synthesisers and voltage control (Beck, 1971). Beck's synthesiser, for example, would incorporate "principles of voltage control and artificial signal production from audio synthesizers" (Collopy, 2014). Sandin describes his own work as "the visual equivalent of a Moog synthesiser" (as quoted in Dunn, 1992). He further states, "I just went through all the Moog modules… and said if you center their bandwidth to handle video and you do the right things with sync, what would they do?" (ibid.) Therefore, the influence of audio synthesisers on the earliest video synthesisers cannot be ignored. The result of this influence and implementation means that the video synthesiser modules could be controlled in the same way audio synthesisers could, with control voltage and crucially, the same control voltage. Automatically, one can already imagine how strategies of audiovisual composition could be achieved with a design like this – running the same control voltages to control the audio synthesiser and the video synthesiser would provide sound and images which are related or synchronised with each other.

The above discussion shows how video synthesisers have a history of inheritance from analogue audio synthesisers. More specifically, the aspect of control voltage and the use of different modules played a significant role in the development of a modular platform for video synthesis, already compatible with modular audio synthesisers for synchronicity. However, this is only half of the story.

2.3 DIGITAL SYSTEMS AND COMPUTER GRAPHICS

While there was significant progress with analogue circuits for video synthesis, the digital movement was also under development around the same time. On one hand, there was the addition of digital circuitry in analogue synthesisers for added control (Collopy, 2004, p. 75) and on the other hand, there was development of complete computer-based systems for graphics generation. John Whitney was one of the pioneers in the field, working both with computer graphics as well as investigating sound image relationships. His concepts would also be based on the writing of Greek philosophers as he developed the concept of "differential dynamics" (Whitney, 1980). To him, the harmony of audio and video

> …assumes the existence of a new foundation for a new art. It assumes a broader context in which Pythagorean laws of harmony operate. These laws operate in a graphic context parallel to the established context of music. In other words, the hypothesis assumes that the attractive and repulsive forces of harmony's consonant/dissonant patterns function outside the dominion of music. Attractions and repulsions abound in visual structures as they become patterned motion. This singular fact becomes a basis for visual harmony with a potential as broad as the historic principles of musical harmony. (Whitney, 1980, p. 9)

In other words, he is also talking about the concepts of congruency and binary opposition, bound together in the tradition of music, with rhythm, structure and intervals. His achievements in developing these notions in a computational system, even though they were not in real time (Levin, 2000), are ideas that are still in practice in contemporary audiovisual works.

Other possible routes for generating graphics through the computer were also being developed, most notably by Ivan Sutherland and Myron Krueger (Golan, 2000, p. 33). Instead of focusing on the relationships between audio and video, Sutherland was working towards emulating natural processes towards drawing (Sutherland, 1964) and Krueger's work would develop connections between interaction and computer graphics (Levin, 2000, p. 33). Collopy (2014, p. 75) argues that like all fields of electronics technology, the digital and analogue realms are very much interconnected. The computer would be an important instrument for precise control and "[t]he digitization of video art was not inevitable progress then, nor did it ever become absolute. Rather, it was a gradual and deliberate experiment that bore fruit in the form of new technological phenomena to explore." (Collopy, 2014, p. 75)

As computers became smaller, more affordable and more powerful as the years went on, the role of computers in the arts kept increasing, and this is true in the field of video synthesizers as well (Collopy, 2014, p. 82). The earlier pioneers of analogue video synthesizers also worked with this emerging platform (ibid.), but computer graphics saw even more development in two other fields – visual effects in films and the video game industry. While the visual effects industry strove for better and more realistic looking graphics, the major concern in the video game industry was real-time graphics generation (Caulfield, 2018). Both of these fields have led to major improvements in the computing technology that we have available today – and plays a huge role in the artistic audiovisual performance world as well with platforms like Unity, a game engine, integrating approaches towards real time filmmaking and performance (Savov, 2017; Unity Technologies, n.d.).

2.4 THE SHIFT TOWARDS DIGITAL AND THE RE-EMERGENCE OF THE ANALOGUE PLATFORM

> "Welcome to the world of the modular synthesiser. An electronic music instrument that infiltrated popular music during the sixties and seventies, was a relic by the 1980's that is now, in the 21st century being reborn in forms both familiar and strikingly new." (Fantinatto, 2013)

Continued development of electronics meant the landscape of electronic music instruments kept evolving. Additionally, the analogue era of modular synthesisers were also expensive and impractical for traveling musicians. Taking advantage of the advances of technology and to solve the problems of cost and practicalities regarding size, Moog redesigned his modular synthesiser into the Minimoog in 1970 (ibid.). Other designers and manufacturers also presented their synthesisers to the world at this time, like the EMS Synthi and the ARP2600. Artists like Kraftwerk were shaping a new musical landscape with their music, using these machines, bringing the synthesiser to the fore in music production (Whalley, 2009). With these improvements, even though they would address the needs of the travelling musician and cost significantly less, it also meant that the modularity of the earlier instruments was being lost in the process.

Fully digital instruments would follow suit. With the development of the Musical Instrument Digital Interface (MIDI) standard in 1981-82 and digital signal processing (DSP), the hardware of music synthesisers were soon flooded with various new instruments. These instruments would not need any sort of patching, relying on digital menus and button combinations instead and would be synchronised with each other using MIDI (ibid.). The Yamaha DX7 and the Fairlight instruments would play a huge role, becoming the instruments of choice for many working musicians (Fantinatto, 2013; Trocco & Pinch, 2009, p. 317). Such synthesisers would also bring to the forefront synthesis methods previously out of reach in analogue synthesisers, like frequency modulation (FM), as implemented in the DX7 (Chowning, 1973).

Aside from musical instruments implementing digital circuitry, computers were also being integrated in the music production workflow at the same time. Even though software for composition and synthesis had existed for a few decades till the 1980s (Puckette, 2002), it was not until the development of software like Pure Data and later Max/MSP alongside the MIDI standard being available in personal computers like the Macintosh (ibid.) that it saw significant improvement to exist outside of an academic and research paradigm. With the development of DAW environments that was to soon follow, the analogue modular synthesisers were losing their significance in favour of a MIDI based and computer based workflow for composition and performance.

While the 1980s would be a dark time for modular synthesisers, they would never fade into complete obscurity (Fantinatto, 2013). A true revival, however, would only happen with the introduction of the Eurorack format by Dieter Doepfer in 1995 (Dalgleish, n.d.; Fantinatto, 2013). The Eurorack format today has become the go-to format for modular synthesisers, both for in terms of new companies developing modules as well as practising musicians working with modular synthesisers. While still relying on the concepts of discrete modules with control voltage compatibility, the Eurorack format has also incorporated modern digital microcontrollers and offers a truly modern modular environment for sound synthesis (Connor, n.d.).

This is, of course, in conjunction with the proliferation of digital audio workstations that are widely used for music production these days ("Global Music Production Software Market to Triple in Value in the Next Five Years," 2018). One important thing to note even in this environment is the rise of virtual modular platforms for making music. These virtual modular platforms, like VCVRack, Softube Modular, Reaktor Blocks, etc., are all based on physical modular synthesisers, with virtual patch cords that need to be connected between modules to make highly customizable audio routings. The philosophy of the control voltage is abstracted into this digital realm (Fasciani & Rahman, 2018) and can be traced back to the foundational principles of how Pure Data and Max function, "communicating with meaningless numbers" (Zicarelli, 1991).

2.5 CONTEMPORARY LANDSCAPE OF VIDEO SYNTHESIS

When it comes to contemporary video synthesis practices, however, this hybrid nature has not really caught up. While companies like LZX Industries are making modular video synthesisers and video synthesis modules, they all follow an all-analogue circuitry, as seen even in their latest product, the Vidiot, released in 2017 (LZX Industries Product Documentation., 2018/2019).

Digital circuitry has been adopted in hardware in digital video synthesiser like the Critter and Guitari ETC, but it lacks the modular nature of integrating with control voltages (even though it has MIDI connectivity) – which limits the possibility of integrating with a high level of synchronicity with audio synthesisers ("ETC," n.d.).

The options on a completely digital platform for video synthesis running on a computer are practically unlimited, however, and has become both ubiquitous in terms of availability and practice (Levin, 2009). A detailed analysis of the contemporary platforms and tools are done in a later chapter but the lack of a modular yet modern physical video synthesis platform is important to note here considering the subject matter of this thesis.

3. METHODOLOGY

This thesis follows a practice-based research methodology. Candy (2006) defines practice-based research as an "investigation undertaken in order to gain new knowledge partly by means of practice and the outcomes of that practice" and this may be demonstrated through various outcomes, including creation of an artefact. Furthermore, practice-based research projects also need to be supplemented with "textual analysis or explanation to support its position and to demonstrate critical reflection" (ibid.). The development of the hybrid video synthesiser instrument is the artefact and this thesis is the supporting document demonstrating critical reflection.

This practice-based research project follows an iterative development process. This process can be broken down into a few cycles of development. Contemporary practices of video synthesis platforms and methods are identified and their weaknesses understood (chapter 4). Design requirements are then formed addressing the same and is followed by a prototyping phase to develop the new instrument and platform (chapters 4 and 5). This prototyping phase itself follows an iterative process with each requirement addressed and then critically evaluated for further improvement and/or refinement. Additionally, performances with this instrument are also carried out, further leading to reflection towards future iterations of the platform (chapters 5 and 6).

The process of developing the instrument has led to re-evaluation and refinement of the thesis question, a common occurrence in practice-based research (Skains, 2018). Skains (2018) notes "there may be significant effects on the composition process of continuing contextual research in theory and creative works" in such a form of research and it holds true for this project as well. He further argues that practice-based research "… is given to exploration and significant moments of discovery, which are largely unpredictable at the start of the project. Thus serendipity can lead to new perspectives on the research, reshaping the project goals throughout." (ibid.) While there has not been any major reshaping of the main aim of the project, there definitely has been new perspectives that have been gained towards the refinement of the required functionality and is reflected in the iterative development phase of the project.

This written thesis is the final stage in the development of a working version of the instrument and serves the role of the final stage of practice-based research demonstrating "substantial contextualisation of the creative work" (Candy, 2006). Skains (2018) argues that in addition to serendipitous connections gleaned during the creative process, argument formation and exegesis are the final step towards a practice-based research methodology. This written thesis fulfils that stage, providing analysis and insights gained towards the creation of the instrument and using it for live performances (chapters 7 and 8). Skains (2018) further notes that post-textual analysis demonstrating deeper understanding of the domain also leads to call for future research and investigation. Chapters 7 and 8 provides this as well and takes into consideration future directions that need to be investigated towards further development of the instrument.

In terms of a timeline, the initial prototyping period for this project was during October 2017 to March 2018, where the hardware was developed. The basic software architecture was also implemented during this time. The first performance using this instrument was in early March 2018, following which a hardware revision was required and multiple iterations of the software followed towards better usability as a platform that could be used live towards audiovisual performances. The final performance at the point of writing this was in September 2018 and the written analysis and reflection, serving as post-textual analysis and exegesis, is what follows, in the form of this thesis.

4. ANALYSING CONTEMPORARY AUDIOVISUAL PRACTICES TOWARDS FORMING DESIGN REQUIREMENTS

For drawing up granular requirements towards the design of the instrument to fulfil the aims of the thesis, it becomes important to identify the features that characterise an audiovisual instrument. Furthermore, an analysis of the contemporary approaches is also required towards understanding the gaps that need to be addressed. This chapter discusses the same and defines a framework for understanding the contemporary practices. Finally, a set of requirements for the design of my instrument are defined.

4.1 CLASSIFYING AND IDENTIFYING CHARACTERISTICS OF AUDIOVISUAL INSTRUMENTS

As discussed in chapter 2.1, based on the ideas of Chion (1994) and Grierson (2005), an audiovisual composition strives towards achieving "added value". Grierson (2005) identifies audiovisual congruency and binary opposition towards as the primary means to such a goal (see chapter 2.1). In the contemporary realm, there are various methods and practices that can be adopted towards such a goal.

First of all, one could look at whether the same tool or platform is being used towards composing both the audio and visual material. Using platforms like Max/MSP/Jitter, one could compose both sonic and visual material tightly bound together and perform with the same. Such an idea could be extended to Ableton Live for audio composition alongside Max4Live, which is an instance of Max/MSP/Jitter that is tightly implemented within the Ableton DAW. On the other hand, the sonic and visual worlds could exist on separate platforms, for example, a musician or DJ using a platform meant for audio alongside another platform meant solely for visuals. Both of these approaches have their advantages and disadvantages and therefore such a distinction does not lend too well towards analysing the end goal and judging it against whether it contributes towards added value or not. Therefore, it becomes important to define the characteristics that constitute an audiovisual instrument or platform.

Franco et al. (2004) discuss the issues for defining a platform suited towards real-time audiovisual performance, and according to them, there are a few characteristics that define a flexible audiovisual platform for such a goal. This instrument, they argue, should have the following features:

1. Real-time (improvisatory) performance capabilities for the creation of images and sound.
2. Compositional structures, event organization and modification.
3. Expressiveness.
4. Mapping flexibility between audio and visuals.
5. Modifiers and effects.
6. Learnability.

Levin (2000, p. 53) also proposes similar ideas, discussing audiovisual instruments with the following qualities:

1. The system makes possible the creation of dynamic imagery and sound, simultaneously in real time.
2. The system's results are inexhaustible and extremely variable.
3. The sonic and visual dimensions are malleable.
4. The system permits the performer to create or superimpose their own ideas over conventions and idioms of established visual languages.
5. The basic principles of operation are easy to deduce while sophisticated expressions are possible and mastery is elusive.

Put simply, both Franco et al. (2004) and Levin (2000) are discussing ideas of customization, flexibility and complexity that can be achieved through such an instrument. Even with these overarching characteristics, it would be quite demanding to provide an exhaustive list of all the available systems, platforms and software that are used for audiovisual performances to analyse. In practice, two different classes of instruments and platforms exist and can be classified into 1.

Artist-developed tools for performances and 2. Leveraging existing solutions towards building a performance.

When it comes to artist-developed tools, through Franco et al. (2004) and Levin's (2000) classifications, these solutions provide the highest flexibility and adherence towards an audiovisual performance system as this provides the artist absolute control over every aspect of their (audio)visual material. This however, comes at the cost of the initial investment of time and gaining expertise in their role as a developer first versus immediately being able to compose artistic material. Therefore, it becomes a matter of interest to note the ease of use of such an approach versus having the range of complexity that is offered and is discussed in detail through the concept of affordances and constraints in a later section of this chapter (see chapter 4.2).

When it comes to existing tools geared towards video synthesis for audiovisual performances, many options are available. These range from hardware manufacturers like LZX Industries, Critter and Guitari to software tools, often targeted as VJ software, like Resolume Arena, Modul8, VDMX, etc., to standalone software video synthesisers like Lumen. While a VJ software does not usually correspond to the definition of a video synthesiser as they are primarily video loop playback based systems, they do still provide a platform for audiovisual performance and they also afford means of analysing audio in real time to synchronize visual material with. Additionally, allowing mapping flexibility to any MIDI based hardware control surface, the control interface can be open ended as well. These are often the choice for large scale concerts in the pop and electronic dance music (EDM) genres and cannot be ignored from the discussion. However, production of audio and video for these kinds of compositions are usually separate and while they are still audiovisual performances, they do not correspond to what is defined as an interactive audiovisual system as defined by Franco et al. (2014), nor are they compatible with the ideas explored by Grierson (2000), etc., as the audio and visual elements exist as separate entities.

A distinction can be made in terms of the platform as well. Hardware tools, like the LZX offerings, the 3TrinsRGB+1 or the CHA/V are analogue hardware solutions for video synthesis which can be used for audiovisual performance while any computer based software or framework, be it commercially available or developed by an artists can be termed as digital. Additionally, any tool that integrates both physical and digital approaches can be called hybrid offerings. The approaches and complexity for such can vary widely as they could be both artist developed or off the shelf.

4.2 AFFORDANCES AND CONSTRAINTS IN INSTRUMENT DESIGN

Magnusson (2010) analyses approaching instrument design through the concept of affordances and constraints. While his paper addresses digital musical instruments, the same ideas can be extended towards audiovisual platforms and purely visual tools as well. Discussing using existing solutions versus developing their own, he writes:

> Problems with the former lie in the conceptual and compositional constraints imposed upon users by software tools that clearly define the scope of available musical expressions. It is for this reason that many musicians, determined to fight the fossilization of music into stylistic boxes, often choose to work with programming environments that allow for more extensive experimentation. However, problems here include the practically infinite expressive scope of the environment, sometimes resulting in a creative paralysis or in the frequent symptom of a musician-turned-engineer. (ibid., p. 62)

This same phenomenon is seen in purely visual performance systems, both in hardware and software. This is where instrument development becomes important and decisions need to be taken with regards to how much affordance to grant to the performers. While a full development environment offers the most affordance, the constraints put into the instrument often lead to more immediate output. Therefore, it is important to find the right balance between the two.

To such an end, Magnusson introduces the idea of mapping as design constraints (ibid.). The core idea discussed in his paper introduces the musician interacting with a virtual or physical controller which uses a mapping engine connected to the sound engine for performance and composition. Through this interaction and sonic and haptic feedback (and additionally, visual feedback), the musician can continue their process of composition and performance.

A video synthesiser or an audiovisual system can be analysed through a similar model. In contemporary practices, various controls are provided to the performer which are mapped to various parameters of the visual engine for real time manipulation. With regards to developing an instrument enabling a performer to do the same, three things become important – the control interface, the mapping engine and the visual engine, aspects that will be important for me to address when designing a new platform.

4.3 PROPOSING A FRAMEWORK FOR ANALYSIS

My proposed framework for analysing contemporary approaches towards audiovisual performance therefore relies heavily on the above discussion of both the key characteristics of an audiovisual performance platform alongside its ease of use, or in other words, its affordances and constraints. Additionally, I would also like to propose a few further points of consideration towards such an analysis making my key points of inquiry into:

1. Ease of use and complexity of operations that is offered towards audiovisual performances,
2. Economic viability, and,
3. Adherence to modern technology standards.

4.4 ANALYSING AUDIOVISUAL SYSTEMS THROUGH THE PROPOSED FRAMEWORK

With the various distinctions in mind, I can analyse the existing systems with the properties I have mentioned above to provide a clearer picture of the current offerings in place.

4.4.1 EASE OF USE, CUSTOMIZATION, AND COMPLEXITY OF (AUDIO)VISUAL MATERIAL

This ties in with both the ideas of design constraints (Magnusson, 2010) as well as the properties presented by Franco et al. (2004) and Levin (2000). The better the possibility for customization or complexity or mapping strategies that are possible on a platform, the less easier it is to use, generally speaking. For example, one can just connect a screen to any LZX video synthesiser or open up Resolume to be able to display graphics and have control over various parameters to build a composition or performance and therefore it is easy to use with the barrier of entry set low. Frameworks like openFrameworks or Max/MSP/Jitter do not afford that to the artist, the artist has to build a logic for video synthesis themselves. But, with the former, the customizations and complexity that can be achieved are much lower than when developing a system on the latter platforms.

This is also true for audiovisual mapping, as publicly available tools only allow basic audio analysis of features like amplitude and frequency, making it difficult for audiovisual synchronicity beyond rudimentary parameter mapping. The simplest examples of these can be found in any audio playback software equipped with a visualizer, even though the above mentioned tools can go a bit further than that. The commercial tools also usually are standalone video output platforms which means the audio composition and performance is happening on a different software or platform. This leads to further gaps in integrating the two. This is where artist developed tools really shine as they can control each and every aspect of how audiovisual material is generated, played back, mapped or analysed.

The level of complexity can also be distinguished between analogue circuitry based video synthesis platforms versus digital platforms. While the analogue platform has its own distinct aesthetic, generally speaking, truly complex video synthesis is only possible on digital platforms.

4.4.2 ECONOMIC VIABILITY

Commercially available tools, obviously, need to be purchased. While the ranges can vary, they are usually more expensive than using open source tools or frameworks towards building their own instruments. Balancing the cost is the fact that commercially available tools are usually more stable environments and see continued updates keeping up with developing technology like operating system upgrades in a software based approach.

4.4.3 ADHERENCE TO MODERN TECHNOLOGY STANDARDS

Display technology is advancing every passing day. It is not uncommon to see high resolution displays at low costs being available. To be able to support such higher resolutions, newer technology standards for providing output is needed. Hardware based on analogue circuits output the video signal through the S-Video or VGA standard, as seen on the LZX offerings, the CHA/V or the 3TrinsRGB+1. While not uncommon, they are slowly being phased out for digital standards like an HDMI signal. Digital, computer-based applications have no limitation on what kind of video signal it can output. This is often important to keep track of as performance venues equipped with projectors and screens are only compatible with a limited set of inputs.

Through the above analysis, various advantages and disadvantages of the contemporary offerings can be noticed. For the design of my implementation towards a hybrid video synthesis platform, I wanted to address as many of the disadvantages as I could while retaining as many of the advantages as possible. The next chapter discusses the requirements towards such an instrument and then my implementations towards the same.

4.5 DESIGN REQUIREMENTS BASED ON ABOVE ANALYSIS

As noted in the overall aims and objectives in chapter 1, I wanted to leverage the digital and analogue tools I was working with towards building a video synthesis platform. Essentially, it meant that I was attempting to develop a hybrid video synthesiser – a physical, tactile and modular instrument with a digital brain. Towards such a goal, more granular requirements needed to be defined based on the above discussion.

First of all, I wanted my platform to be compatible with the Eurorack format. There are a few different reasons for that. First of all, I was already working on music production in the Eurorack format. As discussed earlier, modular video synthesis platforms of the past have had a rich history of inheriting from the modular synthesis platforms of their time and I wanted to follow a similar path. This automatically gave me a few advantages. First of all, with compatible voltage sources, I would be able to drive both audio and video together towards fulfilling the characteristics identified by Franco et al. (2004) and Levin (2000). Secondly, even though my focus was audiovisual performances, I would only need to develop the video synthesis part as the Eurorack modular format already offered modules toward sound synthesis, composition and performance. Finally, as I would be working with established standards, I would not need to create a system from the ground up.

Furthermore, I wanted this video synthesis platform to be both easy to use as well as extensively customizable towards generating complexity.

Additionally, the platform needed to adhere to modern technology standards.

Lastly, I wanted the barrier of entry to be low, meaning, primarily keeping the costs low.

5. DESIGN AND IMPLEMENTATION

5.1 INITIAL PROTOTYPING STAGE

I would classify the initial prototyping period as the development period that started with designing the hardware interface and the initial software framework till the fabricated printed circuit board (PCB) and the first performance. For clarity, I am breaking down this section between the hardware development process and the software development process, even though they were being developed simultaneously.

In terms of architecture of this system, the following block diagram breaks down the different components that were originally planned.

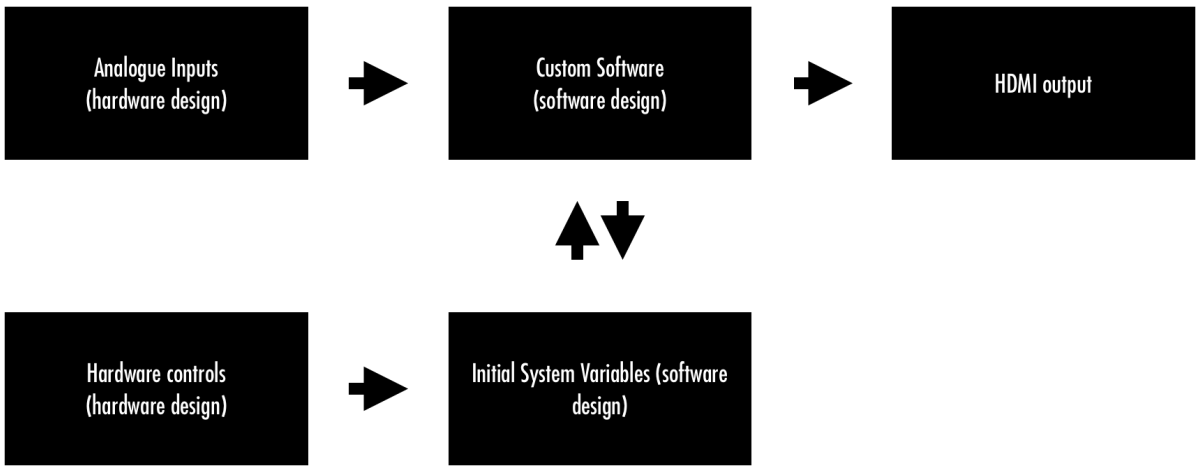


Figure 5: Planned system architecture

Analogue inputs refer to CV addressable parameters which are either in conjunction or in parallel to hardware panel controls that would drive the main visuals software. Ultimately, it is output as a HDMI signal, adhering to modern technological standards.

5.1.1 HARDWARE DEVELOPMENT, DESIGN, AND IMPLEMENTATION

As noted in the requirements in chapter 4.5, the hardware had to be compatible with the Eurorack standard. This essentially meant two things, 1. Being compatible in a physical form factor with Eurorack modules and 2. Being compatible with the various CV sources that exist within a Eurorack system. Additionally, I had to choose the digital microcontroller that would be used to develop the platform on, which would also need to be economically viable.

In terms of physical size, the Eurorack standard, as defined by Doepfer as the A-100 format is 3 rack units high with variable widths for each module as required, measured in horizontal pitch (hp). The depth is undefined and is only constrained by the various Eurorack enclosures that are available from many different manufacturers. Eurorack enclosures are most commonly available in 84hp (the width of a standard 19" rack) or 104hp and the depths can vary anything between 40 mm to more than 10 cm.

5.1.1.1 RASPBERRY PI AND POWER SUPPLY REQUIREMENTS

For the digital microcontroller platform to develop my instrument on, I chose the Raspberry Pi, for multiple reasons. First of all, being both small and relatively cheap, it fulfilled the requirements of fitting in the Eurorack size as well as being economically viable. Moreover, while there are a few different microcontrollers that are in use today, like the Arduino, Teensy or Bela boards, none of them have a display output. The Pi on the other hand, apart from just having a display output, has it through an HDMI port, making it adhere to modern technology standards. Furthermore, it runs a version of Linux, which makes it easier to develop software on, with native compatibility with the open source C++ library openFrameworks, which I was planning to use. Lastly, it even has a 40-pin general purpose input output (GPIO) interface, making it easier to expand with custom hardware.

Another thing to consider while working within the Eurorack standard is the power supply. Eurorack modules are powered through a busboard with a standardized 16 pin power header which has +12V, -12V, ground and 5V rails available ("Technical Details A-100," n.d.).

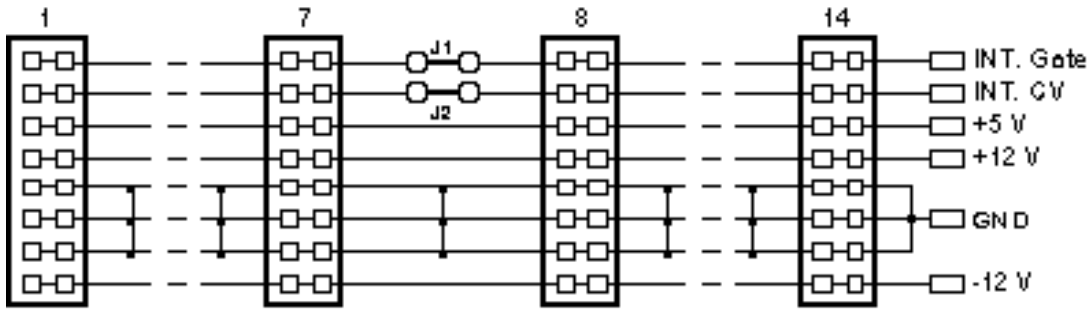


Figure 6: Eurorack power bus

As the Raspberry Pi requires 5 volts of power, I initially wanted to run the Pi directly off the Eurorack power rails. However, it soon became apparent that that was not going to be feasible. The Raspberry Pi's power consumption can go up to 1200 milliamperes and most Eurorack power supplies do not provide sufficient current for such a load – both through the 5 volt header or through the 12 volt pin, regulated to 5 volts. I decided to power the Raspberry Pi through an external USB cable with this in mind. I knew I would still have to access the Eurorack power rails though, for the additional circuitry that had to be developed to successfully interface the Pi with the range of CV sources in the Eurorack system. Therefore, there is an additional power connector on the developed hardware which uses a standard Eurorack power header. It is not used to power the Raspberry Pi in any way but only used towards powering the different integrated circuits (ICs) that are used in the design so that the hardware is compatible with the full range of Eurorack voltages. Please refer to the circuit schematics and the board design included in the open source release and linked in the appendix to see how it is implemented.

5.1.1.2 INTERFACING WITH CV SOURCES

Even though Eurorack modules run on bipolar voltages between -12 volts to +12 volts, the ranges of CV are much more limited. Doepfer defines three different kinds of voltages within the

Eurorack system, audio signals, control signals and trigger signals ("Technical Details A-100," n.d.). As the essence of a modular synthesiser lies in using control signals and trigger signals to affect parameters, my design concerns the same. Control signals are defined as voltages ranging between -2.5 volts to + 2.5 volts by Doepfer, but more commonly seen in Eurorack modules to be in -5 volts to +5 volts range. For envelopes, it is usually 0 – 8 volts. Triggers, or gates (which are trigger signals of longer duration), which are used to sequence events, are unipolar and while the ranges vary, a rising edge detection is often used to respond to such events.

For control signals, there were two challenges to solve. First, the Raspberry Pi is not equipped with an analogue to digital converter (ADC). The most common solution to input analogue voltages into the Raspberry Pi, and the one I eventually adopted, is to use the MCP3008 IC ("MCP3004/3008 Datasheet," 2008). The MCP3008 is a 8-channel 10-bit analogue to digital converter chip that can communicate with the Raspberry Pi over the SPI protocol. The second challenge was to interface with the bipolar range of analogue signals as the MCP3008 with the Raspberry Pi is only rated to convert analogue signals between the range of 0 and 3.3 volts.

There are two approaches that are used with microcontroller based modules with CV inputs to solve this particular challenge. Modules like the Radio Music from Music Thing Modular (Whitwell, 2014/2019) equip the incoming pins with under and over voltage protection and they effectively only respond to the 0 – 3.3 volt range of incoming CV. While this is a valid approach, this leaves a wide range of resolution in the incoming CV unusable, even when attenuated to the usable range. The other approach is to *scale* the incoming CV range to a range that is usable by digital circuits. This is commonly seen in the Mutable Instrument's range of modules (gillet, 2013/2019). An incoming range of CV, usually -5 volts to +5 volts is scaled to the usable 0 – 3.3 volts that digital circuits are compatible with. This is the approach I decided to adopt as well.

As the design and schematics of all Mutable Instrument range of modules are available open source, it is a good reference point to see how they handle this CV scaling. They use rail to rail operational amplifiers with a reference voltage of -10 volts to scale the CV to the usable range.

While taking cues from their design, I tweaked the approach somewhat so that I could use regular, inexpensive operational amplifiers which would not need a -10 volt reference voltage source. Figures 7 and 8 illustrate the differences.

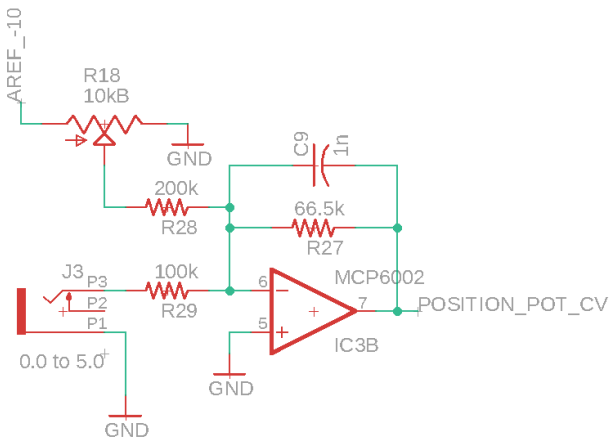


Figure 7: Mutable Instruments, CV input circuit, Clouds

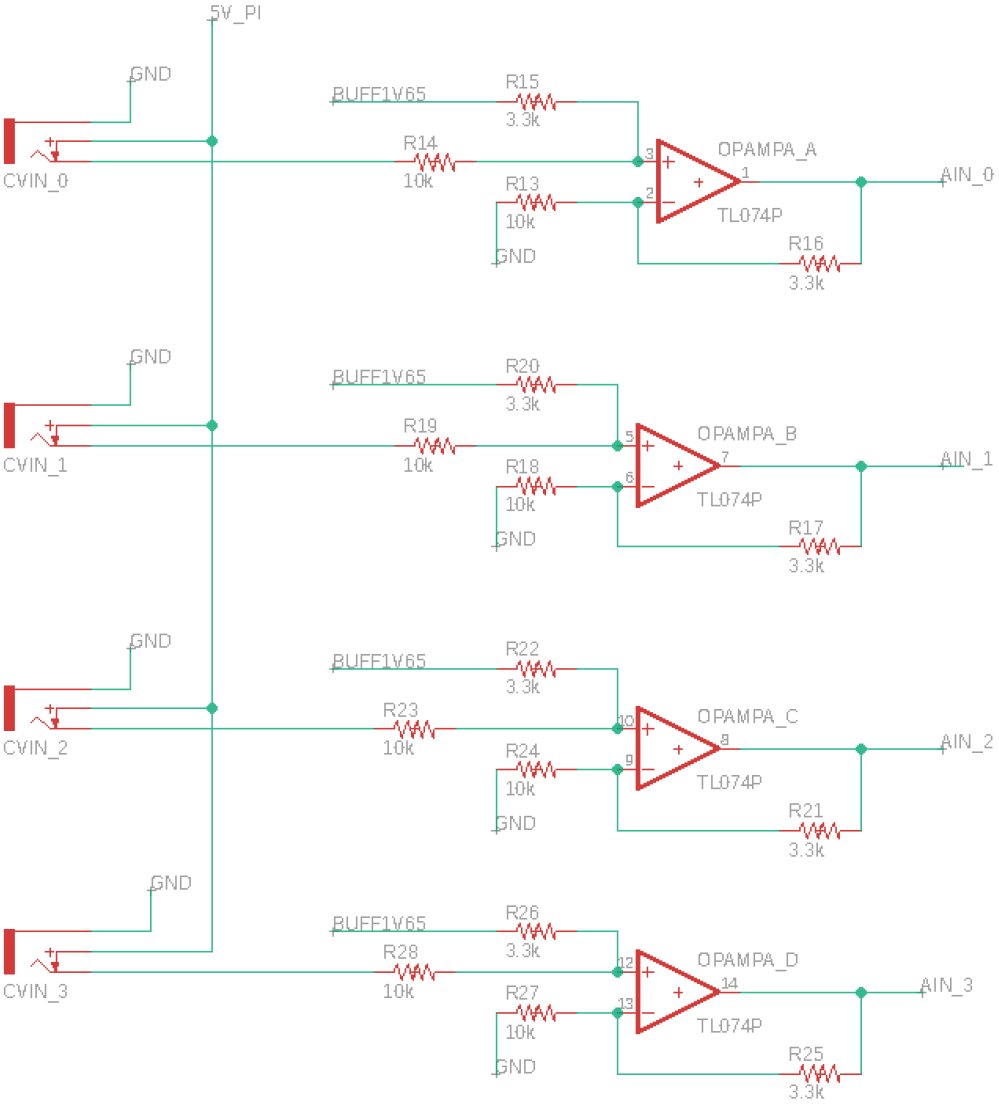


Figure 8: Analogue input section in my design

Another thing to note in the schematics is the use of the Schottky diodes for over and under voltage protection. With these in place, if for some reason the voltage scaling failed, the voltage reaching the IC would still be in the range that could not harm the IC. This is also a common approach, regularly seen in the designs of many Eurorack circuits, the Music Thing Modular modules serving as an example.

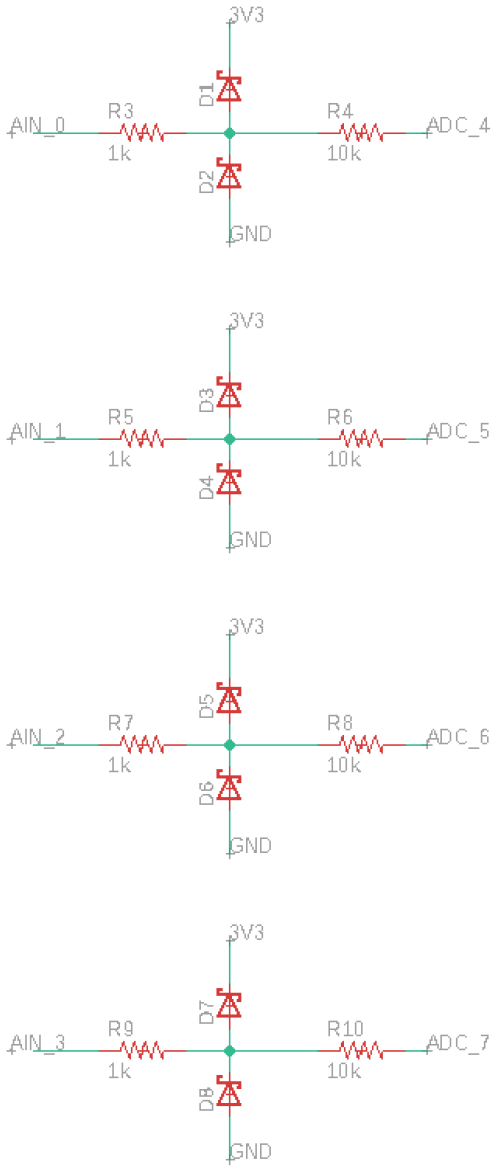


Figure 9: Diode protection on ADC input section

5.1.1.3 PANEL CONTROLS

CV sources expand the functionality of existing panel controls which are available through knobs, faders and switches and are used to set parameters either in the absence of CV inputs or in conjunction with them. Having hardware panel controls was also an important addition to make. One common method of implementing it is with a voltage divider circuit with a potentiometer with the variable output voltage being read by the analogue to digital converter in microcontroller based modules. In pure analogue modules, the voltages can be directly fed into circuits. When this knob also offers CV connectivity, the knob itself becomes an attenuator, or in some cases, attenuverters, for the input CV. One of the ways to implement this feature is to have switching jacks, where the jack is normalized to the positive voltage rail going into the voltage divider circuit and plugging in CV breaks the normalization and the voltage divider circuit further divides the incoming CV. In my case, I had two ways to deal with this. I could follow the similar strategy, but that the voltage divider circuit would need to be powered from +12V (or +5 volts) and go through another voltage scaling circuit and into the ADC.

However, this approach presented a few problems due to the varying voltage levels I was dealing with. Additionally, I was expecting the incoming CV as a value between -5 volts and +5 volts – and this presented problems with regards to which voltage rails to use and normalize the voltage divider to and how to scale them. As the MCP3008 chip offered 8 analogue inputs, I decided to split it into two sections of four inputs, the first set as CV inputs and the second set as voltage dividers using the 3.3 volts from the Raspberry Pi. With reading these 8 channels I could then pair them on the software side with each other to handle attenuation. With this approach, I also had to utilize the same switching jack behaviour, but instead of the traditional approach, they are normalized to 5 volts in the absence of CV inputs as this gives me the highest value after CV scaling and pairing them with the voltage dividers works as intended.

5.1.1.4 GATE SIGNALS

Trigger signals in other words are momentary gate signals and they are simple on-off states that the system can identify. Similar to the knob and CV control over continuous parameters, I wanted to implement this too in a way that it could be set manually by the user and overwritten by incoming gate signals through patch cables.

Some of the same challenges as documented earlier presented itself with regards to voltage levels. In theory, I could accept any level of gate signal voltage and provide under and over voltage protection on the inputs. Paired with switching jacks, I could add panel control switches as well. Even with this type of implementation, however, some other considerations had to be made. The first was determining how many of such gate signals I wanted to have available on the panel. This would be further dictated by the availability of the number of GPIO headers which accept digital inputs.

Researching the schematics for open source and/or publicly available circuits, a common feature became apparent – the use of a comparator based circuit for incoming gate signals. A comparator circuit compares an input signal against a reference voltage and outputs an on or off state which can be reliably read by a digital input pin. Using a comparator in the middle before

going into the digital pins, it would be possible to control the voltage going into the pin and to be able to set a threshold for the incoming voltage coming in and being able to reliably detect a rising edge. Comparator circuits can be designed either through an operational amplifier ("Op-amp Comparator and the Op-amp Comparator Circuit," 2015) or through a dedicated comparator chip like the LM290X. As I had some of the former available immediately when I was prototyping this circuit and none of the latter, I tried to implement it using those.

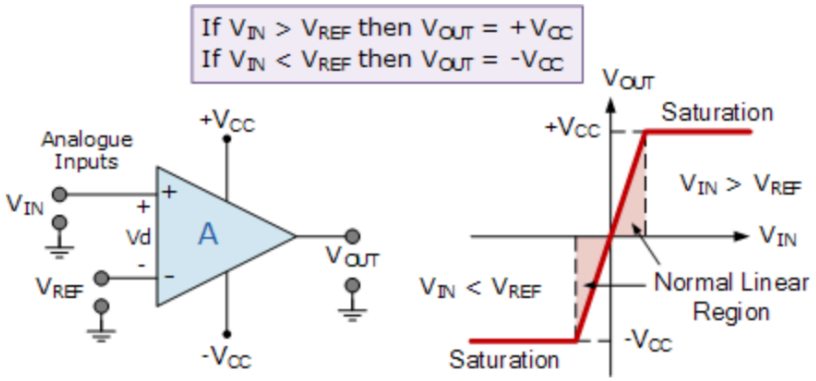


Figure 10: Operational amplifier-based comparator (ibid.)

This particular approach did not end up working reliably in my testing, however, and I eventually redesigned this section with a more traditional comparator chip. These chips have four comparator circuits in them and I decided to implement two of these chips giving me eight gate inputs. I also decided to threshold level at 2.5 volts. This was for two reasons, firstly, if gate signals or low frequency oscillators were going to be at high at 5 volts, it was half of that and secondly, it was easily derivable from the 5 volt rail through a voltage divider circuit by using two resistors of the same value. The comparator would return a 3.3 volt result if the incoming gate signal was high which was safe for the Raspberry Pi and because I was powering the comparator chip off the 12 volt rail, I would not need to consider overvoltage protection in the inputs of the comparator. However, the comparator chip does work on unipolar voltages, so I did add protection against negative voltage.

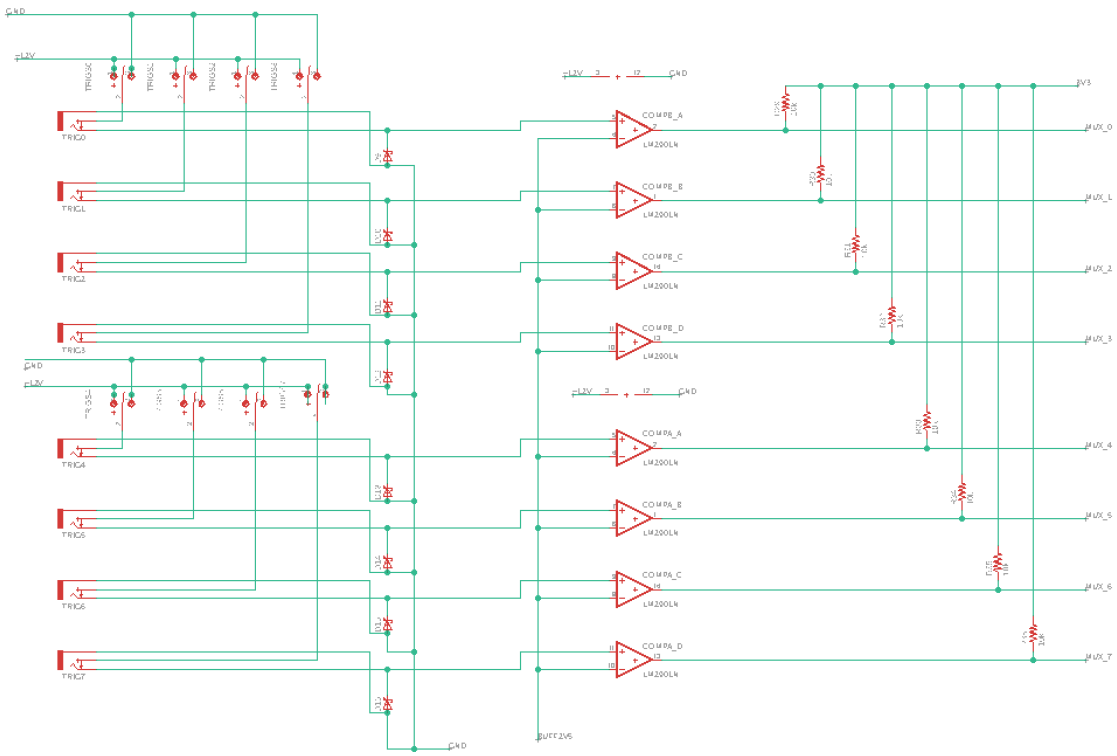


Figure 11: Implementation of comparator-based gate inputs

5.1.1.5 MULTIPLEXING

Instead of reading all eight gate signals separately and utilizing eight separate GPIO pins, I decided to use a multiplexer circuit using the CD4051 (“CD405xB Datasheet,” 2017) integrated chip to read the states. A multiplexer takes in multiple inputs and outputs only one of them through its output, the selection determined by its channel selector inputs. Apart from saving a few GPIO pins, this route also provided me with a more elegant way of reading the states through software. Link to full schematics are included in the appendix.

5.1.1.6 LED DRIVER

Another key feature in performative instruments is to have a visual feedback for the current state of configuration. I was planning to use the eight gate signals as means for achieving compositional structures through selection and sequencing different visual algorithms and

therefore I wanted to have light emitting diodes (LEDs) to provide me with visual feedback of the current state of configuration.

This is implemented with Ken Stone's LED driver schematic ("Ken Stone's Modular Synthesizer," n.d.). The only difference between his design versus mine is that I used a chip which has eight transistors in it instead of eight discrete transistors. Using a transistor based design like this makes sure that LEDs are not drawing power away from the gate switches and inputs which can cause less voltage going into the multiplexer causing unpredictable behaviour and also keeps the LEDs themselves isolated from the rest of the circuit. LEDs draw a fair amount of current which can cause problems in circuits and driving them with transistors avoid that. Link to full schematics are included in the appendix.

5.1.1.7 FROM BREADBOARD PROTOTYPING TO PCB FABRICATION

The aforementioned circuits were all prototyped on a breadboard first and due to space constraints, each circuit was developed as a single channel of application. I developed and tested one channel of CV scaling, one channel of gate input, etc. The breadboard itself was powered from a Eurorack power supply and I could test the circuits with Eurorack modules. The Raspberry Pi was connected to the breadboard through a GPIO breakout cable. Relevant measurements were made and protective measures undertaken before connecting anything between Eurorack modules and the Raspberry Pi GPIO pins.

Once each aspect of the circuit was developed, it was a matter of duplicating them. I wanted to fabricate a PCB and build my video synthesiser as a Eurorack module, which meant I had to design the circuit in a software for fabrication as well as design a panel to mount it on.

I used EAGLE to design the circuits for fabrication. The process in EAGLE is twofold, first the schematics are laid out and secondly, the board is designed and routed. The most commonly used parts for Eurorack already existed as an EAGLE library (Whitwell, 2015/2018), which meant

that I did not have to design any footprints for any of the parts I was using – most of the non Eurorack specific parts are also rather common parts.

While drawing out the schematics was straightforward, as I designed the board, I also had to consider parts placement and how they would be used on the front panel. Alongside that, I also had to make efficient use of space such that the module itself would not take up too much space on a Eurorack rig. There were multiple trial and errors when it came to designing the board layout and the eventual layout consisted of CV inputs on side with its relevant knobs and the gate inputs, with their switches, on the other side alongside the LEDs. For routing the traces, I used the auto-router built into EAGLE. While considered bad practice by some, routing traces is a specialised skill that I lack and I was not designing anything too critical which would be affected by how the traces were routed.

Once this process was complete, the files required by a fabricating house were generated and the board sent for manufacturing.

The fabricated PCB (figure 12) was received a few weeks later, the components soldered and then tested towards its functions. This took place a few days prior to the first scheduled performance using this instrument, and is documented further along this thesis.

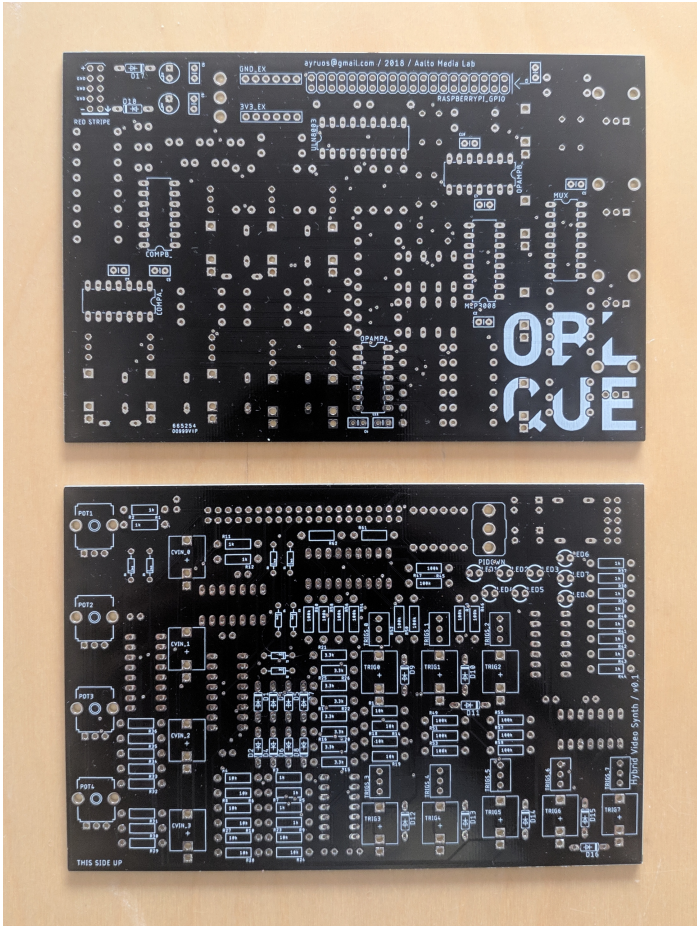


Figure 12: Fabricated PCB

5.1.2 SOFTWARE IMPLEMENTATION

The process of software development started alongside the hardware prototyping. This can be broken down into two stages, first, the software framework for interfacing with the hardware was developed and following that, the software development for the visuals engine was undertaken. As the Raspberry Pi runs a version of Linux, there were various choices with regards to which programming language or framework I could use for developing the software. For this purpose, I chose openFrameworks.

openFrameworks is an open source creative coding C++ toolkit and is a popular platform for media artists to develop their projects on. openFrameworks provides a few key advantages over other languages and frameworks based on python, etc. First of all, it is cross platform, which meant I could write and test the code not only on the Raspberry Pi but on other platforms, like a regular computer as well. Secondly, openFrameworks is also built with visual output in mind, wrapping platform specific graphics frameworks within the library. As my primary goal was visual output, this was an important consideration. Finally, I was already working with openFrameworks for various projects at this point and had familiarity with regards to its syntax and structure. Therefore my familiarity with the platform was an advantage.

5.1.2.1 OFXGPIO AND HARDWARE INTERFACING

The first key part to develop was interfacing an openFrameworks application with the Raspberry Pi GPIO pins and the hardware I was developing. While the core library of openFrameworks does not contain such features, the ofxGPIO addon (Longobardi, 2015) extends openFrameworks to do so. It is also well documented and provides example code for the MCP3008 ADC, which I was using as part of my hardware, and therefore was easy to implement. For reading the multiplexer, and through it, the gate inputs, a simple for-loop is used, switching the multiplexer addresses to read the relevant signals.

```
//reading ADC values
for(int i = 0; i<chip; i++){
	analogIn[i] = a2d.getValueAllChannel(chip)[i]
}

//reading gate inputs through multiplexing
int r0 [] = {0, 1, 0, 1, 0, 1, 0, 1};
int r1 [] = {0, 0, 1, 1, 0, 0, 1, 1};
int r2 [] = {0, 0, 0, 0, 1, 1, 1, 1};

for(int i = 0; i<mux.size(); i++){
	gpio14.setval_gpio(ofToString(r0[i]));
	gpio15.setval_gpio(ofToString(r1[i]));
	gpio18.setval_gpio(ofToString(r2[i]));

	gpio17.getval_gpio(mux[i]);
}
```

5.1.2.2 SOFTWARE ARCHITECTURE

After successfully implementing the code interfacing with the hardware, the software architecture needed to be developed. I decided to keep the visuals framework separate from the hardware interfacing application, choosing to do so with modularity in mind. Keeping the visual framework separate from the hardware would allow it to be developed independently and if required, could be completely re-engineered with a different application or even technology without having to redo the hardware interfacing part of the application. It would also allow testing and development of the visual framework on any other system which did not have the hardware interface available, which was a further advantage.

I used the Open Sound Control (OSC) protocol for the two separate applications to communicate with each other. The hardware interfacing application was modified to add two streams of data,

the CV and the gate signals being sent out over separate channels. The visual framework was set up to receive these values first and then the logic of operations was implemented.

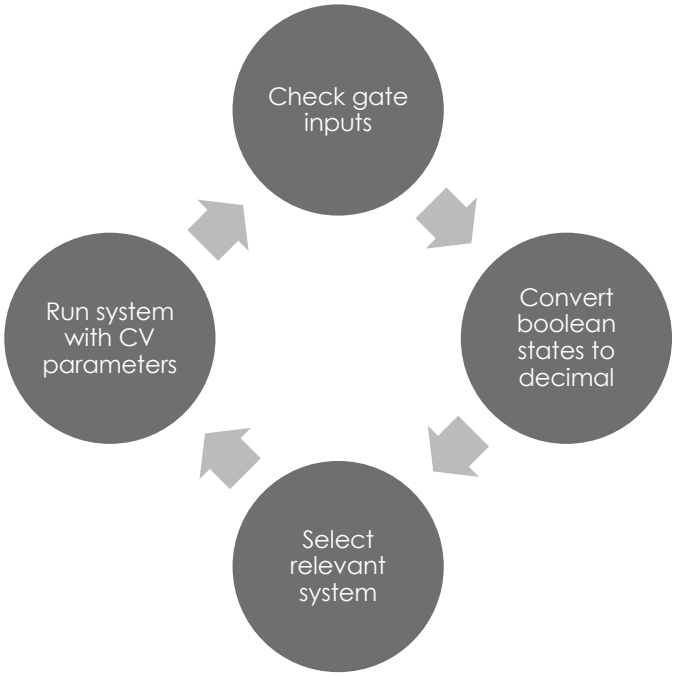
Regarding the logic of operations, the CV was intended to drive parameters in the visual algorithms while the gate signals were intended to be used towards selection and sequencing between different available algorithms. For this, I abstracted the idea of having different 'systems', with their own individual 'subsystems' with additional ways of overriding them with 'modifiers' or 'effects'.

The eight gate inputs, with that logic in mind, are split up into groups of three, two and three respectively. The first three are used to select the 'system', the second two, the 'subsystems' and the last three as 'modifiers' or 'effects' existing over and above the selected system-subsystem combination.

The logic of switching between the 'systems' and choosing their 'subsystems' are based on binary-to-digital conversion of the gate inputs. With three of the gate inputs, eight different combinations are possible to switch between the 'systems', the further two gate signals allow four more combinations and that would decide the 'subsystem', leaving the last three gate inputs free to override the selected 'systems' and 'subsystems'. The overrides are inspired by my own workflow as a VJ allowing enabling/disabling the output, inverting the output colours and drawing white strobes. Simple operations like this have always allowed me to add in real time improvisations on top of the visuals in my VJ practice and something which I have felt made visual performances more dynamic. Adding similar functionality in this video synthesis environment therefore felt natural to me.

Once the visual algorithm was decided by the state of the gates, the visual algorithm would then draw on screen with the four CV inputs driving some parameters.

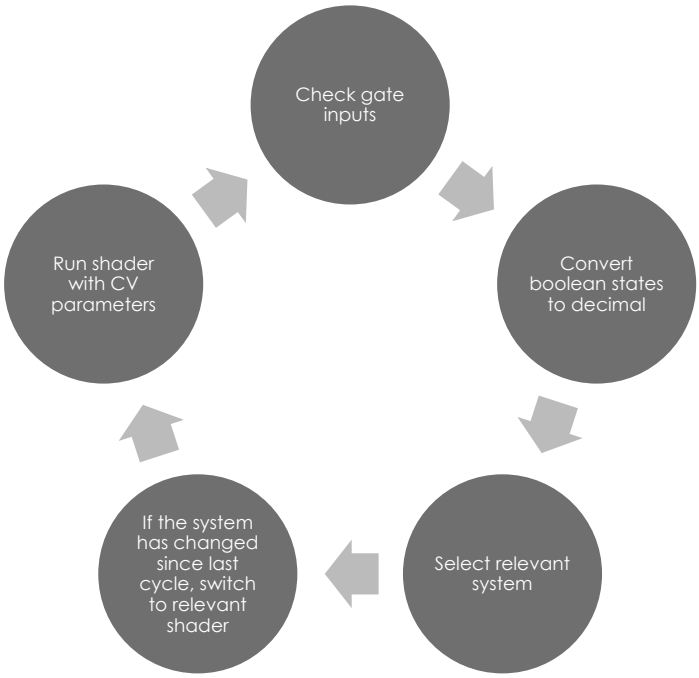
The following figure demonstrates the update cycle of the application based on the above description.



5.1.2.3 IMPLEMENTING VISUAL SYSTEMS WITH FRAGMENT SHADERS

There are various methods to draw visuals on screen using code. openFrameworks also makes it possible to easily implement many of these. The various choices include directly manipulating pixels, building interactive systems using particles and such, algorithmic drawing using basic shapes and finally drawing using pixel shaders. There are advantages and disadvantages to each of these methods. Pixel based methods can be used to directly manipulate every pixel that is drawing to the screen. Building interactive, autonomous and semi-autonomous systems allow different dynamics and behaviours. Different algorithms can be used to construct patterns and layouts using simple shapes, which openFrameworks makes available to draw with simple methods. However, there are two fundamental problems in using all of these aforementioned methods, especially in the use case I was implementing them for. First of all, these methods are all highly CPU intensive. The Raspberry Pi CPU is not very powerful and I was not sure how it would handle systems like this. But the main reason that I did not attempt investigating any of these approaches is the fact that these are usually implemented at the application level using C++ which are then compiled and run. This meant that if anyone wanted to make changes in the

future, they would have to directly edit the source files. Exposing both the core files and expecting everyone who wanted to customize the visuals to be comfortable with compiling an application was the wrong route to take towards making the instrument both easy to use while keeping it extensible and customizable. The last option, using fragment shaders, would mean that the shaders would reside on disk as separate files which could be easily edited or replaced without needing to recompile the application seemed like the better choice to make here. Based on the above logic, the update cycle can demonstrated with the following figure.



Fragment shaders are small pieces of code that run on the graphics card, and unlike drawing with the CPU, they process colour values of each pixel on screen parallelly (Patricio, n.d.). This makes the program highly efficient and it is possible to generate complexity with very little computation compared to using code on the CPU side.

openFrameworks has an implementation of shaders using the ofShader class and uses what is known as the shader pipeline internally by default as well. Therefore it is relatively simple to implement custom shaders through reading files from disk. With this approach in mind, I implemented the logic of shader file selection depending on the current 'system' in use.

Shaders get parameters from the host application through the use of *uniforms*. Uniforms are read-only parameters that is received by all the parallel processing cores in the graphics processing unit (GPU). Therefore the CV values are passed onto the shaders that is being drawn to screen using the use of these uniforms. Before passing on the uniforms, they are also scaled to an usable range and the inputs are attenuated with the potentiometers in the code. Refer to the open source release as linked in the appendix for full source code.

As the focus was more on the development of the instrument and the platform than the actual visuals themselves, I did not develop too many custom shaders for the project but rather relied on open source shaders that are available on platforms like shadertoy.com, glslsandbox.com, etc., to test the performance and response of the instrument.

Developing the software framework with successfully implementing shaders and drawing them to screen as well as receiving the printed circuit board concluded the first prototyping stage and was followed by a performance at Ääniaalto 2018 (see chapter 6.1). Through the lessons learnt during the aforementioned performance (as noted in chapter 6.1.2), certain improvements were necessary and the development period after this point can be termed as the second phase of prototyping.

5.2 PROTOTYPING PHASE TWO, TOWARDS A PERFORMANCE READY INSTRUMENT

The second phase of prototyping from March 2018 onwards consisted of both making the technical system work as intended as well as improving application functionality for further ease of use and extensibility of the instrument. This phase, apart from developing the instrument further, also included three performances and one presentation. As with the overall project, these improvements followed an iterative development process, but for clarity they are grouped in

sections and discusses the development process in a non-linear fashion while keeping the discussion of performances and presentation in a separate chapter.

5.2.1 HARDWARE UPDATES

5.2.1.1 PCB REVISION

In terms of hardware, the first objective in this phase was to address the gate input section of the hardware. The issue with this section on the PCB was the fact that having had prototyped two different ways of implementing the comparator functionality, the schematics I had drawn up on EAGLE was a confused version of both. I tested the circuit on the breadboard again, and making sure it worked, I updated the board on EAGLE. I also took this opportunity to update the footprint of the switches I was using in the first version of the instrument – I had originally intended to use mini switches but they were harder to source and I replaced them with a more commonly available version instead. Having had the experience of the system restarting during the Ääniaalto performance, this was an issue that also needed to be investigated, but unfortunately, I could not replicate the issue in testing. I dismissed the issue as arising from instability of the gate input circuitry, and having updated the design for that section, I sent the redesigned PCB for fabrication.

I received the redesigned PCB a few weeks later and having soldered the components, the hardware was functioning as expected. However, while using the instrument for longer periods of time, I kept experiencing the same rebooting issue intermittently. This was hard to troubleshoot as it was hard to replicate – finding the pattern of when the Raspberry Pi was exhibiting

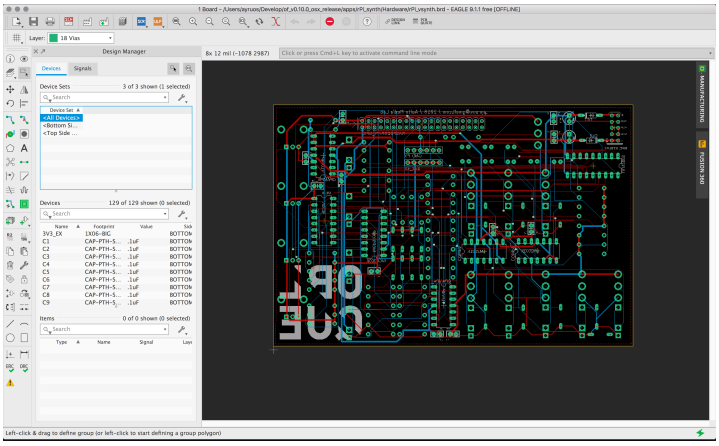


Figure 13: PCB revision

this behaviour seemed random at first. After various permutations and combinations of incoming signals and patching the cords in various ways, it finally seemed like the Raspberry Pi was restarting only when the other end of the patch cord was already plugged in to a CV source. When it was not, there were no issues. Upon further testing, I was finally able to replicate the same behaviour by applying any voltage to the power rail of the Raspberry Pi – and thus it seemed like the in-built protections of the Raspberry Pi were somehow being triggered when a patch cord was being plugged in. While I have no evidence of the same, my hypothesis is that the switching jacks (which were normalized to the 5 volt power rail of the Pi) were somehow not switching cleanly when an external CV was being plugged in.

By cutting the trace that connects the 5 volts rail of the Pi to the switching jack and thus disconnecting the connection, this behaviour was immediately fixed, somewhat proving my hypothesis. However, I still needed the switching jacks to be connected to 5 volts power so that setting the values through the knobs would work properly. As a temporary fix, I attached a 5 volt regulator from the 12 volt power rail and used that as a source to the switching jacks default behaviour. This temporary fix is still in place and I have not had the issue of the Raspberry Pi randomly restarting at any point afterwards. A more permanent fix will be undertaken for the next redesign of the PCB.

While redesigning the PCB, I also designed a panel for it so that it could be mounted in a Eurorack case. In terms of the layout, the different ideas driving the software architecture played a role in designing both the PCB and the panel layout. As illustrated in figure 14, the CV inputs are on the left hand side with their relevant panel controls. The systems, subsystems and effects override inputs are laid out in their own groups with switches and the LEDs are put together in a fashion that it is easy to see at a glance the current state of the gate signals.

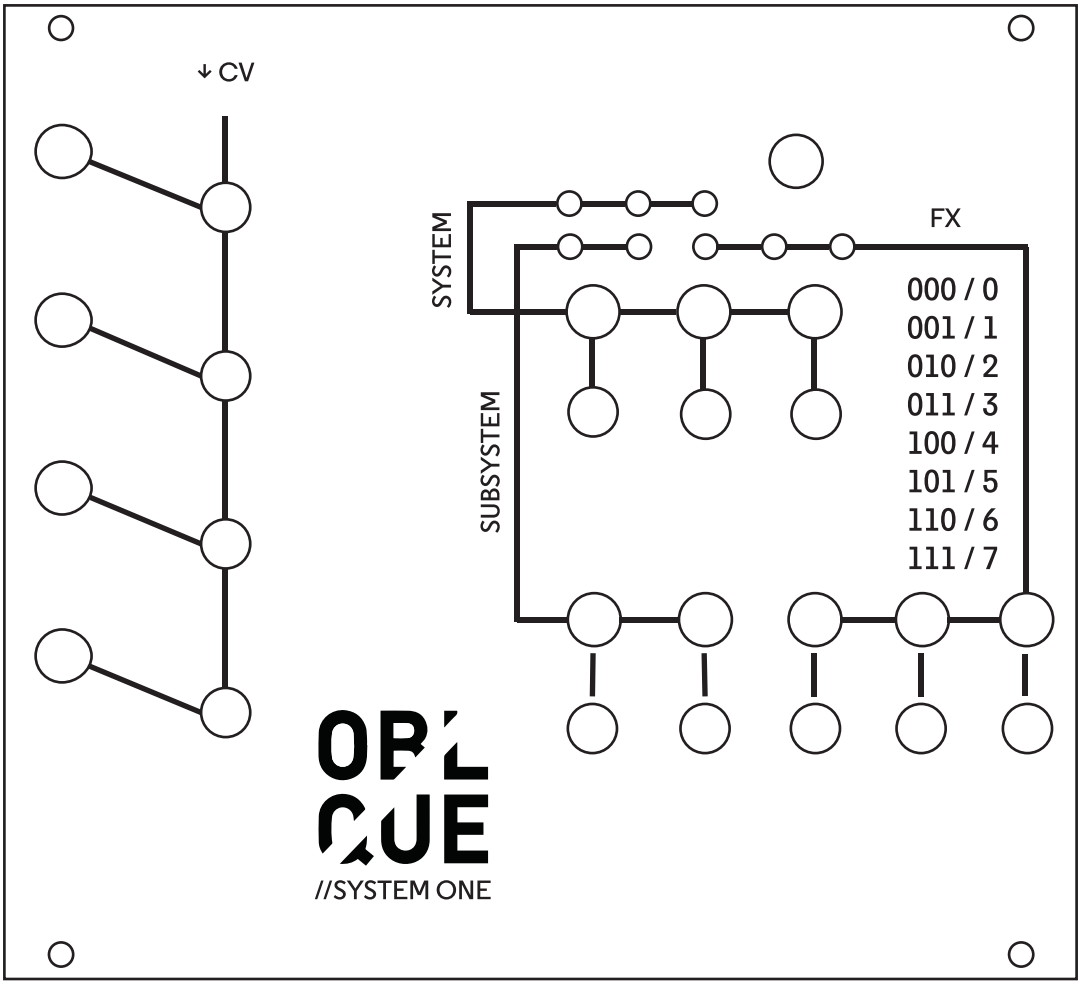


Figure 14: Panel design for the instrument

5.2.2 SOFTWARE ITERATIONS

5.2.2.1 THE CONTROLVOLTAGE CLASS

A new software feature that was required became apparent both through the use in the first performance as well as during testing the visual algorithms, which was finer tuning of reading the analogue control voltages. Part of it was due to the ADC chip which lead to noisy values and there are various methods to deal with it ("Three Methods to Filter Noisy Arduino Measurements," 2017). To have a standardised method to deal with analogue inputs, I made a class which would handle the analogue inputs (see appendix for link to source code). The filtered value is a simple algorithm that relies on the last measured value and changes depending on the filter threshold, defined as alpha and uses the following formula:

```
double ControlVoltage::filteredValue(double _in, double _lastOut){
    double answer;
    answer = _lastOut + alpha * (_in - _lastOut);
    return answer;
}
```

The ControlVoltage class also has a few more methods to deal with the varying outputs of modules in Eurorack systems as well as how different modules respond to them. The first method scales the values exponentially. This is due to the fact that the raw control voltage, depending on the source, would be linear but a lot of musical applications respond exponentially, like scaling pitch for example, and making the visuals respond in a similar fashion makes it have a more immediate relation to the music. I did not use a true linear to exponential mathematical formula, though, and used a common method that is used in musical applications, raising a normalized value to a power of itself towards building up dynamics (Farnell, 2010, p. 413).

The second method handles the input range of the control voltage. While the Doepfer specifications state LFO ranges lie between -5 volts to +5 volts, and which is the input ranges my circuit was handling, a few modules in the Eurorack format use cycling envelopes as LFOs and only have an output in the unipolar domain. While it is easy to offset the ranges using dedicated modules, I still wanted to be able to deal with such inputs without having to go through another intermediate module. Therefore I implemented a method to only respond to a unipolar 0 volts to 5 volts range as an added functionality. With both these functions in place, the code can return the relevant value through the following code block:

```
double ControlVoltage::getFiltered(){
    double answer = filteredValue(currentValue, lastValue);

    lastValue = answer;

    if(bipolarVolts == false){
```

```
            answer = ofMap(answer, 0.5, 1.0, 0.0, 1.0, true);
        }

        if (exponentialReturn){
            answer = pow(answer, 2);
        }

        return answer;
    }
```

All the methods mentioned above can be set up in code in the main file like so:

```
CVin[4].setExponential();
CVin[6].setUnipolar();
CVin[7].setUnipolar();
```

By default, CV inputs are set to be linear and bipolar and the they need to be explicitly set to unipolar or exponential responses as demonstrated above.

5.2.2.2 DESKTOP TESTING ENVIRONMENT

As mentioned earlier, one of the reasons for choosing openFrameworks was due to the fact that I could do cross platform development. Having the ability to develop cross platform meant speeding up the workflow drastically as recompiling for even the most minor change on the Raspberry Pi is often a very slow process. However, two problems presented themselves towards this ability. First, I was lacking the voltage inputs on other platforms and secondly, while openFrameworks and C++ works the same on different target platforms, shaders need to be customized for the graphics hardware that they are run on. Having a cross platform compatibility was important, however, and therefore I had to solve these two problems.

The solution to the first problem was simple. As the main visuals program was responding to values sent over OSC from the hardware interface, I set up another application for use on the desktop operating system which would do the same and have a graphical user interface that could be used to set values mimicking the hardware. The synth_CVTester application (see appendix for link to source code) handles this.

The second issue was not so straightforward. Computers, depending on the graphics hardware, use openGL shaders of various versions. Raspberry Pi's and other ARM based devices like Android and iOS devices use a slightly different standard known as the GLES or EGL standard. While a fair amount of the syntax is similar, there are some differences, however, especially when it comes to the headers of the shaders. For example, openGL shaders start with #version 150, or the relevant version number and EGL shaders start with *precision highp float* – defining the floating point precision of the shaders in use. There are some other differences as well, like how vertices and textures are handled.

After a conversion on the openFrameworks forum ("GLES/Raspberry Pi shaders on desktop? - arm - openFrameworks," n.d.) and understanding where the differences lie between the shaders on the different platforms, I was able to implement a system that could handle cross platform shaders – and that is currently in use in the instrument as well. Essentially, the headers are dynamically loaded depending on the platform the shader is being run on and constants are defined which work cross platform by referring to their relevant shaders. A separate example is provided as a repository (link in appendix) and instructions to write cross platform shaders are provided in the README of this instrument's Github repository (link in appendix).

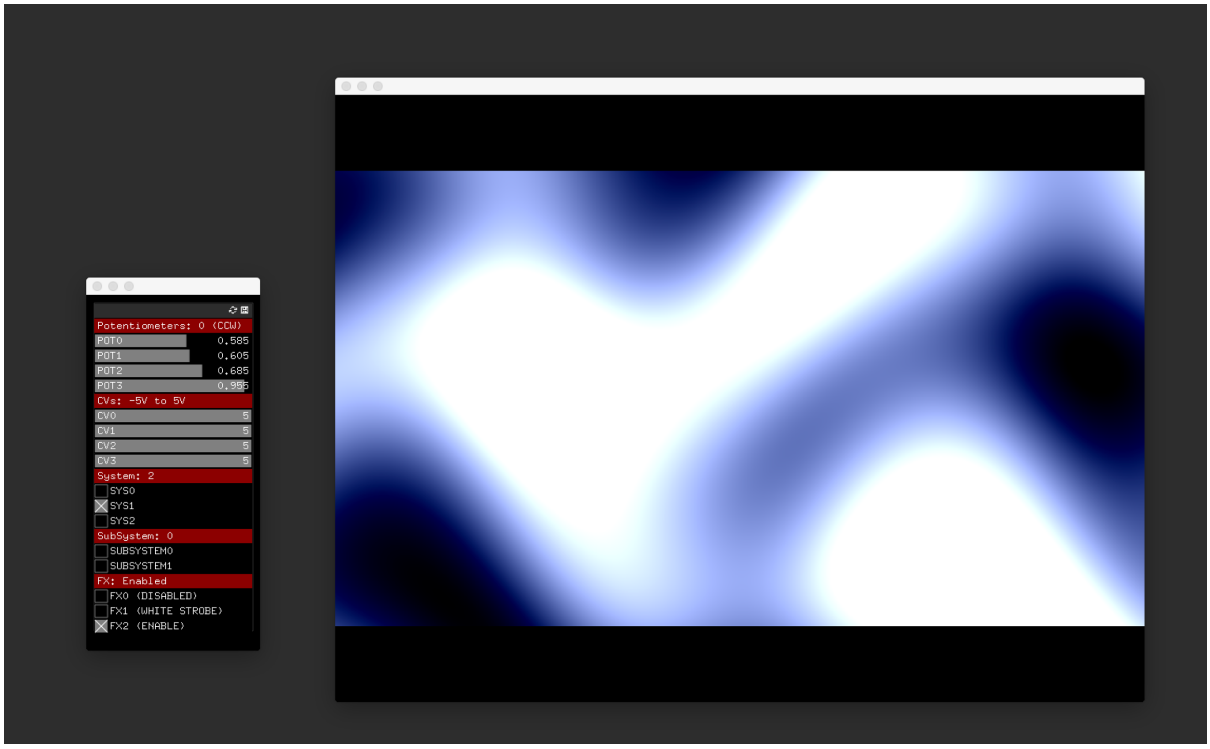


Figure 15: Desktop environment for cross platform testing

With these implementations in place, I could test both the program and the shaders on any desktop platform before syncing it to the Raspberry Pi and compiling it there, which greatly sped up my workflow. I still needed to test on the Raspberry Pi quite often though as the performance of the Pi cannot be compared to the desktop operating system and shaders which worked fine on desktop would often not perform at a decent frame rate on the Pi. Nevertheless, not having to rely only on the Raspberry Pi for testing every additional change in the code was a big addition and it also helps in demoing the functionality of the instrument even with the lack of Eurorack gear, and the actual physical instrument. Additionally, the software-only demo is often far suitable for presenting the functionality of the instrument to an audience who are not well versed with modular synthesisers, as discussed in chapter 6.3.4.

5.2.3 ADDITIONAL FEATURES

5.2.3.1 START UP SCRIPTS

Towards making the instrument function more like a Eurorack module, it needed to directly run the two applications at start up time. For this, I wrote two scripts that would boot the Raspberry

Pi directly into these applications and not require any external keyboard or interface to run the relevant applications.

This approach, however, has a slight drawback. Due to how the Raspberry Pi and openFrameworks work at a hardware level, it requires the display output cable to plugged in when booting up the system.. If the cable is not plugged in, and the projector or screen switched off or not expecting a signal for some reason, the application handling the visuals would crash and the Pi would need a restart. This is not necessarily a huge negative, but the performer needs to be careful in which order the cables are plugged in.

5.2.3.2 COMPROMISES IMPLEMENTED TOWARDS SMOOTHER PERFORMANCE

While drawing with shaders is generally highly efficient and complexity can be generated through rather simple means, it is also true that the Raspberry Pi hardware is not able to handle too complex scenes. Concepts like raymarching are impossible to execute at a decent frame rate. Even some simpler shaders, if it involves slightly more complicated mathematical functions, drawing them at large resolutions is not very efficient.

To make a compromise between performance and complexity, there are certain methods that I had to build into the code. The first of those was to limit the target frame rate at 25 frames per second (fps). While 60 fps is considered a standard in computer graphics these days, for the purposes of live performance which are generally through projectors which rarely run at 60 frames per second, this seemed logical. The second step I took was to draw the frame at a fixed resolution of 1280 x 720 (commonly called the 720p resolution) and then scale to the output resolution, irrespective of whether the target platform was larger or smaller than 720p. Performance projects rarely go above 1920 x 1080 (commonly known as the 1080p resolution) as 4k is still too expensive for smaller scale venues to implement and enlarging 720p to 1080p in a performance setting does not affect the perception of the audience much due to the distance to the screen. While this still does not allow concepts like raymarching to function, it at least lets

slightly more complex scenes run without too much compromise and provides a smoother experience.

5.3 SUMMARY

The design and programming of the instrument went through various iterations throughout the development of the project. The software side is still being modified as and when required while preparing for a new performance or adding new shaders and testing against an always expanding setup of Eurorack modules that are used in conjunction with the video synthesiser.

Even then, there are a few known issues and possibilities for improvement that would be implemented in the near future. The issue regarding replacing the use of the 5 volt rail from the Raspberry Pi to a dedicated voltage regulator as a more permanent fix has already been mentioned. Secondly, it is currently possible that short incoming triggers patched into the gate inputs do not execute expected code, even if the input LEDs light up. This has to do with the fact that the program is refreshing every 25 frames per second and if the trigger is received in between the refresh cycle, it can fail to register. As the LEDs are connected purely through hardware, the visual feedback in this case leads to further confusion. The possible solution for this is to expand the code handling the hardware interfacing to incorporate a queue system for incoming triggers and clearing the cache every time they are executed or read through the main program. The complexity in this approach is maintaining correct timing and length so that the triggers are cleared on time and the execution does not go out of sync.

Lastly, the CV parameters are currently connected to the shaders or the visual algorithms do not follow any particular pattern. For example, in one patch the same CV/parameter could be controlling the overall brightness of the image and in another patch it could be controlling speed of the animation. While a complete standardization is not possible due to the varying nature of the different shaders currently in use, it would be useful to have some sort of standardization so that while sequencing or changing patterns, some sort of parity is maintained with the incoming

CV signal mapping. An incomplete attempt towards doing so has been investigated and have yielded good results, but a wider implementation is still required.

Another additional feature that should be incorporated is being able to set the CV response and scaling behaviour (exponential/unipolar/etc.) through a standard text or XML file outside of the main code. Currently, the program needs to be recompiled every time a change is made regarding how each CV input is handled, and while that is acceptable for my personal use, as one of the goals of this project is to make this as easy as possible for everyone to use, this would be a nice feature to have. Additionally, the instrument could see even more usability improvements by the inclusion of a custom shader uploading tool that could be done through the computer through a web app, for example. As it stands, the only way to edit or upload new visuals is to either access the Raspberry Pi directly or accessing the storage through SSH from another computer – again, not the easiest route for everyone.

In spite of the drawbacks and the improvements that can be made, however, the instruments design and development has led to a stable performance ready module. While there is always scope for improvement, the instrument does adhere to the Eurorack standard, can be used with other Eurorack modules to sync visual elements to sound and is a plug and play solution that is still highly customizable. In the next chapters I will discuss the performances done through the instrument, evaluate the instrument towards audiovisual performance while noting key takeaways and finally, discuss the future directions this project can take.

6. PERFORMANCE CASE STUDIES

This chapter describes the various performances in which this instrument was used for the visuals within the context of audiovisual performances. After the description of each performance, key takeaways are discussed which directly played a role in the development process and further refined functionality of the instrument.

6.1 ÄÄNIAALTO 2018

6.1.1 ABOUT THE PERFORMANCE

The first performance was scheduled to take place at Ääniaalto 2018, in Helsinki. Ääniaalto is a festival of audiovisual arts organized by the student organization of the department of media at Aalto University, DADAry. The 2018 edition of Ääniaalto was scheduled to be held at Vapaan Taiten Tila, a venue for students to organize events. The festival ran for multiple days and my performance was scheduled for 9th March, 2018.

A few days prior to the performance, I received the fabricated PCB. Upon soldering the components onto the board and testing it with a Raspberry Pi running my application, I discovered that the gate inputs were not responding to inputs, nor were the panel mounted switches changing states as expected. Troubleshooting further and comparing the board against my schematics and the prototype on the breadboard, I discovered there were a few mistakes in the schematic that I had drawn up om EAGLE and it required a redesign and re-fabrication of the board to successfully correct. However, not having enough time for such an endeavour before the performance, I had to implement a temporary solution for this issue.

As I did not have a panel for the module at this time, which meant that the video synthesiser could not be mounted in the Eurorack case either way, I decided to use a regular USB keyboard to switch between 'systems' as a compromise in the absence of the gate inputs not functioning. Figure 16 illustrates the setup for this performance.

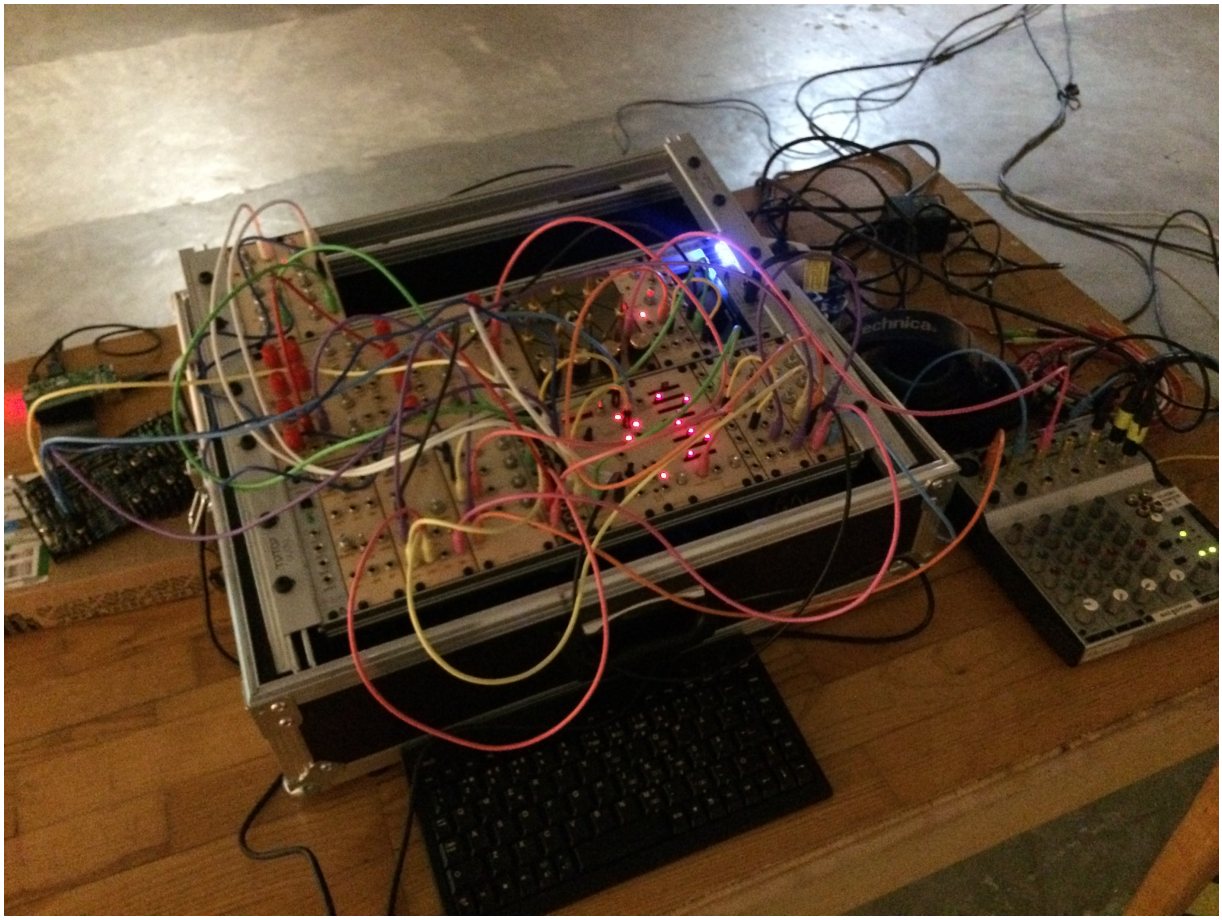


Figure 16: Performance setup, Ääniaalto 2018

While there were certain moments during the performance where I successfully implemented ideas of congruency, binary opposition and correspondences between audio and video, overall, the performance had a major a flaw due to which I would term the performance a failure. The video synthesiser kept restarting at various intervals while patching control voltages into the circuit with no apparent reason. Halfway through the performance during which the instrument restarted several times, I decided to switch the visuals off as the restarting process was distracting and taking away from the performance. It was also impossible to troubleshoot such a behaviour within the scope of a live performance.

6.1.2 KEY TAKEAWAYS

There were a few key takeaways at the end of this performance. Some of them were purely technical in nature, and some were further refinements that needed to be implemented towards better artistic output.

- From a purely functional point of view, the gate input section needed to be redesigned towards their intended use.

- The randomly restarting behaviour needed to be investigated further and solved.

- I needed to rehearse further with this instrument to be able to create more cohesive compositional structures.

- As the range of control voltages were varied, and different modules had distinct response curves to incoming control voltage, there needed to be further flexibility towards how the video synthesiser responded to incoming CV as well. Additionally, incoming CV needed to be slightly filtered in software as well to reduce the slight jitter that the ADC circuit was causing.

6.2 CALCULEITOR PARTY

6.2.1 ABOUT THE PERFORMANCE

The second scheduled performance was organized by Joaquin Aldunate towards testing his virtual modular performance instrument, and I was invited to perform as well, as one of the opening acts (Aldunate Infante, 2018). This took place on the 4th of May, 2018, at Helsinki's Kallio Stage.

This performance used the revised version of the hardware, with the fix for intermittent restarting issue implemented. It also included the panel for the instrument and it was mounted on the Eurorack case as any other module. An edited excerpt from the performance is included in the attached media.

Figure 17: Performing at the Calculeitor Party, 2018

6.2.2 KEY TAKEAWAYS

The takeaways from this performance are varied in nature. Overall, from a functional point of view, the instrument performed without any difficulties, but from an artistic perspective, there were a matter of key issues to look into, some beyond the scope of the instrument per se but more related to the overall ecosystem of modules I was using. More specifically,

- The instrument was stable throughout the performance, i.e., there were no random restarts or odd behaviour.

- I successfully managed to build up compositional structures through the use of sequencers, both in the audio and visual domain, and kept them in sync with each other.

- However, beyond basic sequencing, due to the lack of performance-centric modules in the Eurorack setup that I was using, I did not have the flexibility of improvising beyond building up on the aforementioned sequences.

- While the audiovisual sequences were in sync and could be sequenced in various ways achieving notions of "added value", the entire performance needed to adhere to more traditional musical ideas towards appealing more to the audience.

- I also realised that the decision to develop the video synthesiser within the Eurorack format was a valid one, as a lot of the performance-centric or artistic goals that I wanted to implement towards performing could be easily solved with readily available Eurorack modules.

6.3 LIVE PERFORMERS MEETING (LPM)

6.3.1 ABOUT THE FESTIVAL

Live Performers Meeting (LPM) is a yearly audiovisual festival that was held in Rome for its nineteenth edition in 2018 during 7th to 10th June. It consisted of audiovisual performances, VJ performances, workshops, lectures and a mapping competition, among other things. Apart from an audiovisual performance, I was also scheduled to present my instrument on the preceding day. Unlike the other performances, the audience at this festival comprised majorly of other audiovisual and visual artists.

6.3.2 PRESENTING THE INSTRUMENT

I was presenting the instrument during the day preceding my performance. Instead of a traditional presentation going over the project, its goals and design process, I used this opportunity to make it an interactive presentation. I introduced the project quite briefly and opened up the floor immediately afterwards, inviting the audience to experience using the instrument for themselves and ask any questions that they might have. The takeaways from the

presentation are collated at the end of this section alongside the analysis of the performance at this festival.

Figure 18: Presenting the instrument at LPM, 2018

6.3.3 PERFORMANCE AT LPM

Regarding the performance (edited excerpt attached in accompanying media), learning from the previous performance experience as well as having made further refinements in the software, having added more shaders and having had the time to rehearse more with this setup, I attempted towards less pitched sounds but rather looked at performing more improvised musical events, and driving parameters and sequencing visuals with those events. I also had a few more utilities like attenuverters and polarizers in the rack at this point to further manipulate the CVs going into the video synthesiser. This meant I had finer control over how the visuals were mapped and sequenced. Some of the visuals also had their inputs standardized to an extent, like affecting similar parameters across different shaders and due to that there was a certain coherence that accompanied all the visuals as well.

6.3.4 KEY TAKEAWAYS FROM THE PRESENTATION AND THE PERFORMANCE

Attending this festival as a performer and presenter with an audience comprising mainly of other visual artists definitely contributed a lot towards both understanding the scope of the instrument and analysing my work, as an instrument designer and as a performer. The key learnings are listed below.

- The presentation definitely generated an amount of interest, and while there was live interaction with the instrument and the attendees, the workflow of the instrument was hard to explain and demonstrate to an audience who did not have much experience with modular platforms.

- To elaborate further on the previous point, it was not the video synthesiser that needed to be explained specifically, but rather the modular workflow including the function of the other modules in the case that had to be demonstrated. Modular synthesisers are definitely a niche field and even among composers and musicians working with them, the combination of modules that they have, know and understand varies greatly. Therefore, this video synthesiser would be suited towards persons already working within the modular Eurorack ecosystem, and thus, the target audience would not necessarily be every visual or audiovisual artist in the field.

- To explain the workflow of the instrument better to people without much experience in modular systems and demo it, the cross platform software-only implementation that was primarily intended towards development presented itself to be highly valuable.

- Regarding the performance, however, I was reasonably satisfied. Not only did the instrument perform well technically, I was able to use it towards a flexible audiovisual performance much better than the previous performance.

- I received appreciation from the audience towards the performance as well. Some of the comments included the fact that it was much more interesting to see me perform with a modular system interacting physically with an instrument that was almost alive versus standing behind a laptop leaving the audience with very little understanding of what I was actually doing.

6.4 SOUNDEST LAUNCH

6.4.1 ABOUT THE PERFORMANCE

The last performance with this instrument as of writing this thesis was as part of the launch event of the SOUNDEST zine, published by sound and media artist Ava Grayson ("SOUNDEST Zine," n.d.). The event was held at Oranssi ry, Helsinki, on October 3, 2018 and I performed a 20 minute set.

The main differentiating factor in this performance from the previous ones was the fact that my Eurorack setup had changed considerably in the previous months towards pursuing a more hands on and performative approach towards composing audio. Additionally, I had switched the case I was using to house my modules for better portability. This meant that my main Eurorack setup was a smaller two row one as opposed to the three row setup I was using previously and the video synthesiser was mounted in its own other, smaller enclosure, with a few other utilities specific to the purposes of the video synthesiser module.

As for the performance, the approach did not change drastically when it came to the video synthesis part of the show but had a more organic and live feeling to it instead of being completely sequenced with different modules. Apart from some tweaking in the software, there were no any additional improvements in the software either as compared to the previous performance. However, the addition of touch plate controllers and more performative

sequencers in my Eurorack setup meant that I could play aspects of the performance live and have more opportunities towards live improvisation as compared to the previous performances which relied highly on linear sequencing.

6.4.2 KEY TAKEAWAY

This was a shorter performance and there was essentially only one key takeaway from it - even in the absence of major updates to video synthesiser between the previous performance and this, the addition of newer modules in my setup led to better performance strategies and performability. This further validated my decision of working within the Eurorack format and proved that the instrument was highly expandable within the ecosystem it was being used in.

6.5 PATCHING STRATEGIES

Through the course of these performances I developed a few patching strategies towards audiovisual compositions. Figures 19-22 illustrate some of the same. For clarity, the control voltage sources and the audio modules are abstracted into their core functions and are not restricted to any particular module to provide a clearer overview towards how the video synthesiser can be patched in a modular environment towards audiovisual performances.

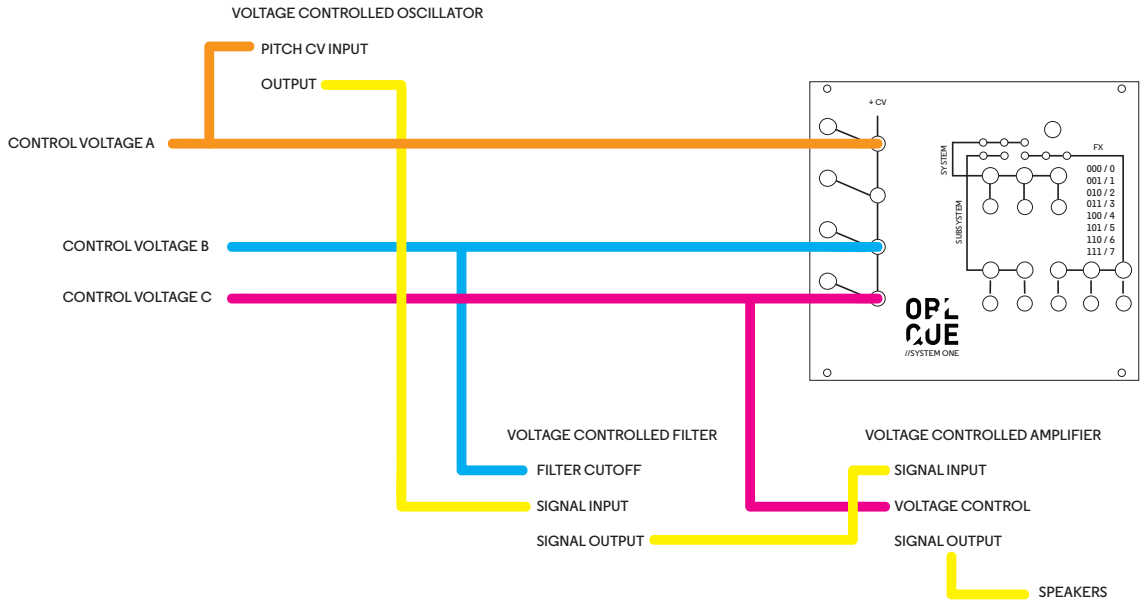


Figure 19: Single system visual parameter mapping with a subtractive synthesis voice

Figure 20: Clock synced visual sequencing (no audio source depicted)

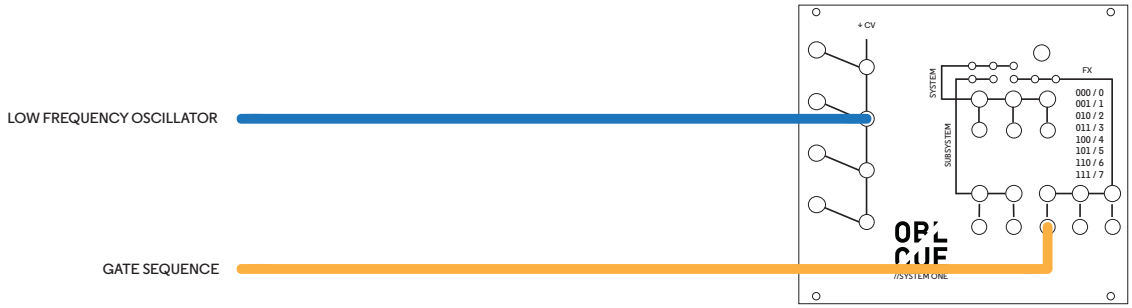


Figure 21: Cycling animation with gated strobes (no audio source or sequencing depicted)

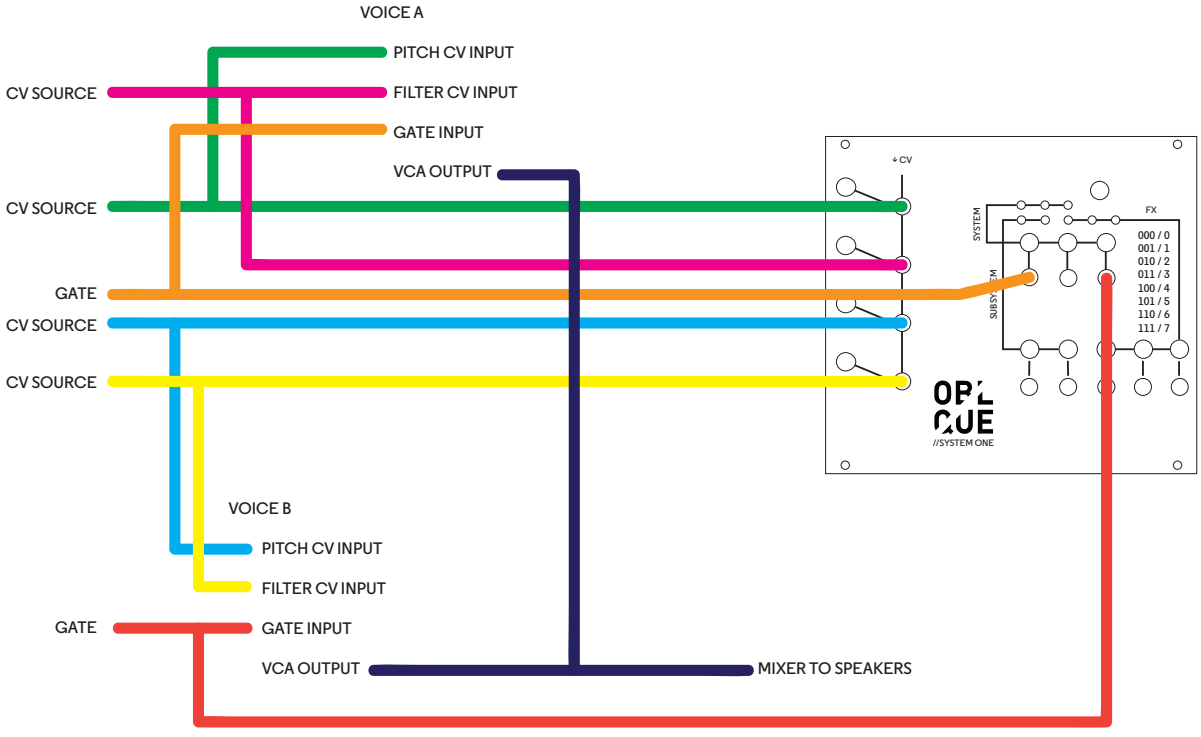


Figure 22: Duelling voices complex patch

7. DISCUSSION

This discussion chapter is broken down into three sections and serves as analysis and contextualisation of the work.. The first section self-evaluates the instrument based on my framework as defined in chapter 4.3 and the design requirements it elicited. The second section provides some observations as learnt through the development process and provides some recommendations for artists working in a similar domain. Finally, the third section discusses the open source release and what it contributes to the field.

7.1 SELF-EVALUATION OF THE INSTRUMENT

As discussed in chapter 1, the primary aim of the project was to build a hybrid video synthesiser for the purposes of audiovisual performance. Towards that end, I took the decision of working within the Eurorack format, analysed the contemporary approaches towards audiovisual performance and set up a framework through which certain design requirements were drawn out (see chapters 3 and 4 for more detailed methodology and analysis). The following section evaluates the instrument based on the same framework and the design requirements the analysis elicited.

7.1.1 EASE OF USE, CUSTOMIZATION, AND COMPLEXITY OF THE (AUDIO)VISUAL MATERIAL

The instrument in its most basic form is quite easy to use. Once powered and connected to a screen or projector, there is immediate visual output that can be controlled in a tactile fashion with the front controls. In this way, it is very immediate and allows for instant gratification in line with many different video synthesisers, both in hardware or software. There is no initial development time required to get an output.

However, it also allows for a high degree of customization and complexity which can be built up via various means. With the ability to patch in external control voltages that allow different means of modification, driving parameters and building up compositional structures is quite easy.

Furthermore, with the way the software is designed, it is also fairly easy to customize or add new visuals to the instrument by modifying the shader files that are stored on the disk.

In other words, based on the features defined by Franco et al. (2014) and discussed in chapter 4.1, the instrument allows for real time improvisatory performance capabilities, can work with compositional structures and event organization, can be expressive via the means of control voltages and offers flexibility towards mapping between image and sound. With the availability of certain modifiers, it is also possible to further add effects and with a simple panel control structure comprising of a few knobs and switches, it is not too difficult to learn as well. A few such strategies are illustrated in chapter 6.5. Therefore, there are enough strategies implemented towards attaining audiovisual congruency or binary opposition, features discussed by Grierson (2005) towards "exploiting added value" with clear design choices made towards affordances and constraints in tune with Magnusson's (2010) discussion, offering a mapping engine with tactile and visual feedback.

However, there are certain limitations as well. While there is flexibility in mapping control voltages to drive audio and video parameters, there is no means of mapping the visual output back to any audio features. Many audio modules allow for features like envelop followers and in a free running animation system, it would further improve audiovisual mapping strategies if I had access to something similar, an overall brightness calculation as a control voltage output, for example.

A fair amount of the evaluation above is based on the assumption that the user already has an understanding of modular synthesisers. As I realized while presenting at LPM 2018 (see chapter 6.3.4), this means that towards an audience who do not have such understanding, the instrument

can appear confusing. Without having other modules offering control voltages and audio sources, the video synthesiser only makes up for one half of an expressive audiovisual performance platform – not necessarily a drawback as building an audiovisual instrument was not the primary aim in any way, but worth pointing out all the same.

7.1.2 ECONOMIC VIABILITY

The instrument is relatively cheap with the estimated cost being under 100€, including the cost of a Raspberry Pi. An exact cost is hard to determine as it depends on the volume of the PCB fabrication and the varying cost of the components between different resellers. However, as no uncommon or hard to find part is used, they are not too difficult to source.

I do not have any plans of selling the instrument as a product or building them in scale. Instead, I have released all the source code, hardware fabrication files, instructions, etc., open source (link in appendix) in the hope that others would find the project useful and would want to build one for themselves.

7.1.3 ADHERENCE TO TECHNOLOGY STANDARDS

On a basic level, the instrument adheres to current technology standards, outputting video through an HDMI port and being compatible within the Eurorack format. This makes it easy to integrate in any modern workflow, unlike some of the other hardware video synthesisers that exist in the market and rely on older video standards. The Eurorack format is also the most popular modular synthesiser platform at the moment with multiple companies offering various modules therefore making the instrument exist within a vibrant community.

7.2 OBSERVATIONS AND RECOMMENDATIONS TO ARTIST-RESEARCHERS IN THE DOMAIN

Through the development of this project, I noticed several points of interest which are listed below. Some are personal observations and some are recommendations to fellow artists and researchers in a similar domain who might be looking towards developing similar instruments, platforms or performance strategies. These have been formed through the development process and provide examples of "exploration and significant moments of discovery" (Skains, 2018) and is a key contribution towards the analysis.

7.2.1 WORKING WITH OPEN SOURCE TOOLS AND CROSS PLATFORM DEVELOPMENT

The open source community at large has been highly helpful towards the development of this project. A lot of the project is developed on top of open source tools and platforms and having access to schematics and circuits of other modules in the Eurorack format is also highly appreciated. However, it was not always been smooth sailing. There is a distinct trend in the industry that can be noticed with competing standards that is making open source and cross platform development difficult. Additionally, with the rapid development of the commonly used operating systems diverging from the core standards that was commonly seen earlier, a lot of the open source libraries are quite often outdated quickly. While discussing those in detail is beyond the scope of this thesis, I would recommend researching the various standards, libraries and frameworks that exist and noting their cross platform and backwards compatibility and considering the adoption and involvement of the community including continued development before committing towards any particular tool or platform.

7.2.2 WORKING IN THE EURORACK STANDARD

Even though Eurorack has a defined standard, it is common to see different manufacturers working within their own ranges of voltage tolerance. For example, most Make Noise modules expect unipolar control voltages whereas most Mutable Instrument modules expect bipolar

control voltages instead. Therefore, the Eurorack standard is quite fragmented in nature. While there are various modules designed towards attenuating and offsetting voltages to use in such a situation, a lot of times there is a fair amount of guesswork involved with regards to what a module is outputting or expecting. This led me towards adding additional code to the ControlVoltage class later (see chapter 5.2.2.1) as I had not anticipated this when starting development of the design of the instrument.

If I was starting out again or redesigning the hardware extensively, I would approach this issue from a hardware point of view – a feature that is commonly seen in physically larger Eurorack modules. Instead of attenuators on the control voltage inputs, it is a better idea to implement attenuverters instead as it allows for scaling and offsetting the control voltage inputs on the panel itself. This does add in more complexity to the hardware design and additionally may take up extra space on the panel as well and raise the manufacturing cost, but is something to deliberate when designing modules within the Eurorack format.

7.2.3 WORKING WITH SMALL FORM FACTOR GRAPHICS PROCESSORS

One of the primary reasons for choosing to work with a Raspberry Pi was due to the fact that it is one of the few small form factor platforms with native graphics processing and video output. However, the graphics performance of the Raspberry Pi left a lot to be desired, and I had to make certain compromises due to that (see chapter 5.2.3.2). Looking at modern devices like smartphones, tablets and even smartwatches, it is obvious that high performance graphics capabilities can be integrated in small form factors. Therefore, it is quite surprising that none of that is available for general purpose use. Again, this thesis is not the place for discussing this issue in detail, but it is definitely a cause of concern that needs to be mentioned. The availability of a general purpose small graphics development platform would not only improve the performance capabilities of this instrument specifically but also open up the field for other purposes. Therefore, when working with the available tools, it is important to be aware of the limitations from the start to have realistic expectations towards technical performance.

7.2.4 WORKING WITH AUDIOVISUAL MATERIAL

Finally, I would like to note a few observations regarding working with audiovisual material. Artistic intent is not the subject matter of this thesis and while no specific visuals have been particularly developed for individual performances or compositions, there still has been some general learnings through the various performances and while testing. First of all, when sequencing between different visual algorithms which have standardized parameter controls, it is important to map the parameters to similar features of the different algorithms. Especially when driving parameters through external CV inputs, this helps in maintaining a certain aspect of alignment between the different kinds of visuals. Secondly, when using visuals that are self-animating, maintaining a parity between the speed of different animations in different visual algorithms contribute towards coherence as well. Finally, when composing visual material, considerations towards the audio material need to be given as well. Though this sounds obvious in an audiovisual practice, as the video synthesiser drives shaders that are inherently separate from the entirety of the audiovisual system, certain artistic intent needs to be considered when working with the visuals in isolation. Put simply, the above three points can be summed towards the recommendation that while a lot of effort can be put towards developing a particular visual algorithm, an overall balance between all the separate pieces need to be maintained, especially when working with different pieces that contribute towards a bigger goal.

7.3 RELEASING OPEN SOURCE AND CONTRIBUTIONS TO THE FIELD

The primary contribution to the field of audiovisual performance in general and to video synthesisers specifically is the development of the instrument and the post textual analysis, this thesis. The instrument has also been released open source (see appendix for a link – the repository is also archived on the accompanying media). This includes all the files necessary for building this instrument – the hardware schematics and board design (including a bill of materials), all the relevant pieces of software (the hardware interfacing framework, the main visuals engine and the cross platform user interface) and a template for the panel.

However, the repository contributes to the community as more than just means towards making this particular instrument. By making the files available open source, I hope to further the development of related projects in the field. Below are a few contributions and extensions that this project makes possible.

- The hardware schematics follow a tradition of open-source Eurorack modules from companies like Music Thing Modular and Mutable Instruments and provides a slightly different approach towards working with analogue control voltages within a digital microcontroller environment, interfacing with gate signals and driving LEDs (see chapter 5). I hope that these can serve as resources to other instrument builders looking to develop hybrid instruments.

- By decoupling the main visual engine and the hardware interfacing software, it is possible to use the hardware and interface it for other purposes or with other software. For example, it would be quite easy to interface with the hardware through a Pure Data patch from the Raspberry Pi and access the current panel settings or control voltages patched in through the relevant OSC channels.

- Similarly, the visuals engine can be used without the hardware interface. A simple extension already exists as part of the cross platform development strategy (see chapter 5), but due to the engine requiring only OSC values, it can be sequenced or interacted with through other software or processes as well.

- Due to the way the visual engine is implemented, it is already quite simple to implement custom shaders towards using them as part of the instrument. Additionally, however, it would be quite simple to extend the concept further and use the instrument either alongside the hardware or standalone as a platform for live coding shaders for visuals.

- The ControlVoltage class (see chapter 5.2.2.1), with its various implementations of filtering analogue inputs, linear and exponential responses, etc., can be used in any

context and is not restricted towards just control voltages unique to this instrument or Eurorack systems.

The above list is just a few ideas on how the material that has been developed for one particular instrument can be used towards other applications, related or otherwise. Therefore, by releasing all the material open source, my contributions in the field go beyond my own artistic work performed and composed with this instrument, the development of this instrument and the written thesis, but contributes to the field in a broader context.

8. FUTURE PLANS AND CONCLUSION

A discussion on the project is provided in the previous chapter towards post-textual analysis .This covers self-evaluation of the instrument, observations and recommendations for others in the domain, and further contributions the development and release of this instrument makes towards the field. However, a project like this is never really complete and there is always scope for improvement, especially when there is personal interest towards use in performance and compositions. Therefore the instrument presented in this thesis can be considered as a prototype version or proof of concept version which will be further developed in the future.

A few known issues and additional nice-to-have features have already been mentioned in the summary of the development chapter and those are the first few things that need to be addressed (see chapter 5.3). Those are improvements towards making the instrument more user friendly and fixing existing shortcomings and do not account for artistic intent. Towards that purpose, further testing in terms of graphics performance is required and also developing custom shaders for composing and performing is required.

Besides this, even though the project has been released open source on GitHub, the instrument has not been formally presented to other artists at large and no user testing has been done either. It would be useful to gather feedback from other artists as well as to investigate how they implement this instrument with their own practices and within their own Eurorack environment. As mentioned multiple times, the ecosystem of other Eurorack modules used in conjunction makes a significant difference towards the performability of this instrument, and therefore seeing it used in other environments would be highly beneficial for developing future versions.

To conclude, however, I would like to note that while the process of developing this project has taken much longer than anticipated in the beginning, with more, smaller refinements that could be further undertaken and a lot of scope for future work, it has nevertheless been a fulfilling journey. Having fulfilled my primary aim of building this hybrid video synthesiser, I now have the opportunity to keep building upon my audiovisual practices through it in a real time modular

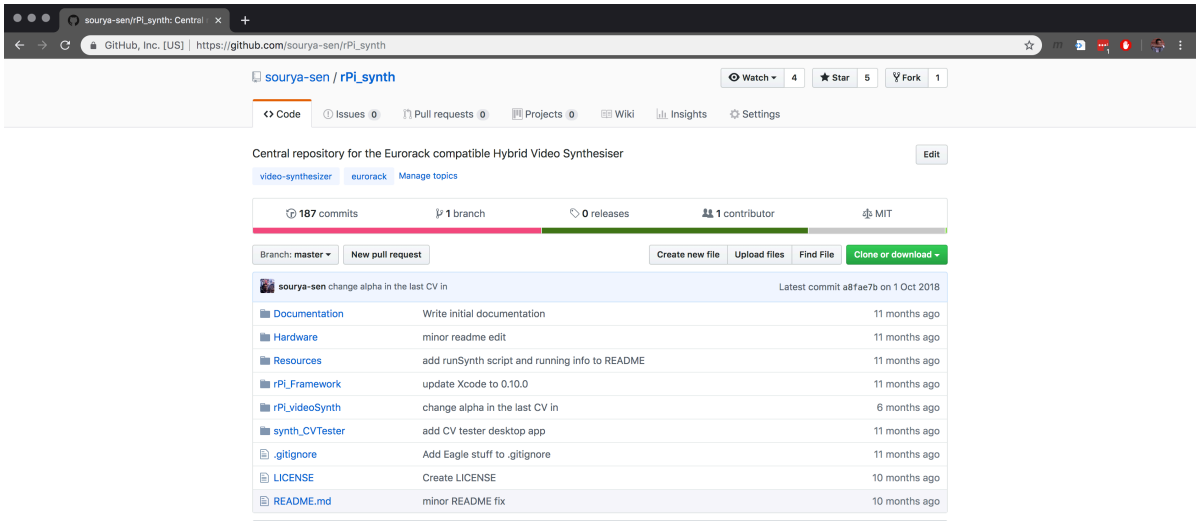
patching environment. While the instrument does have its drawbacks, it also allows enough scope for audiovisual composition and performance.

I am glad that I invested my time in developing a platform like this instead of building a particular performance that could only be used in a specific context. As frustrating as this journey has been at times, it has also been immensely satisfying. Developing a platform is only one half of the puzzle – understanding it and successfully using it is another. Even though I have performed using this a few times, the instrument offers me newer ways of using it every day and I look forward to see what else I can achieve through continued use of this instrument.

APPENDIX

LINK TO GITHUB REPOSITORY AND STRUCTURE OF THE OPEN SOURCE MATERIAL

The open source release can be found online at **https://github.com/sourya-sen/rPi_synth**



The repository contains a README file, explaining the procedure for building the instrument. It provides all the files necessary for the same and the structure is explained below.

The **Documentation** folder contains images used in the readme and also provides a PDF file of the schematics.

The **Hardware** folder contains the .sch and .brd files which can be accessed through EAGLE and are the schematics and board design, respectively. The zipped Gerbers file are the board design exported for fabrication and can be directly sent to any fabrication facility.

The **Resources** folder contains the start-up script that is required to autorun the relevant software on switching on the instrument (instructions on setup are in the main readme). It also contains the cross platform shader headers which are required for implementing the software on different target platforms.

rPi_Framework is the openFrameworks project that is responsible for the hardware interfacing part of the software.

rPi_videoSynth is the openFrameworks project that is the main visuals engine.

Synth_CVTester is the openFrameworks project that runs the Desktop GUI for cross platform development and demoing.

The latest commit of the repository at the time of writing is also included in the attached media.

LINKS TO SOFTWARE AND HARDWARE INSTRUMENTS AND PLATFORMS

There are various software and hardware instruments and platforms that are mentioned throughout the thesis. The following table provides web URLs for the same.

Software/Hardware	URL
3TrinsRGB+1	http://gieskes.nl/instruments/?file=3TrinsRGB1
Ableton Live	https://www.ableton.com/
Arduino	https://www.arduino.cc/
Bela	https://bela.io/
CHA/V	https://jonasbers.com/chav/
Critter and Guitari ETC	https://www.critterandguitari.com/etc
Doepfer Musikelektronik	http://www.doepfer.de/home.htm
Lumen	https://lumen-app.com/
LZX Vidiot	https://lzxindustries.net/products/vidiot
Make Noise	http://www.makenoisemusic.com/
Max/MSP	https://cycling74.com/products/max-features
Max4Live	https://www.ableton.com/en/live/max-for-live/

Modul8	http://www.garagecube.com/modul8/
Music Thing Modular	https://musicthing.co.uk/
Mutable Instruments	https://mutable-instruments.net/
openFrameworks	https://openframeworks.cc/
Pure Data	https://puredata.info/
Raspberry Pi	https://www.raspberrypi.org/
Reaktor Blocks	https://www.native-instruments.com/en/products/komplete/synths/reaktor-6/blocks/
Resolume Arena	https://resolume.com/
Softube Modular	https://www.softube.com/modular#/
Teensy	https://www.pjrc.com/teensy/
VCV Rack	https://vcvrack.com/
VDMX	https://vidvox.net/

CONTENTS IN THE ATTACHED MEDIA AND LINK TO ONLINE ARCHIVE OF THE SAME

The attached media contains:

1. Edited excerpts from performances, Calculeitor Party and LPM 2018
2. Copy of the open source release
3. Additional media, containing images and shorter clips, from prototyping and performances

The same material can also be accessed at **http://bit.ly/sen_thesismedia**

REFERENCES

Aldunate Infante, J. (2018). *Modular composition environment: A tool for improvisation of conventional electronic music.* Retrieved from https://aaltodoc.aalto.fi:443/handle/123456789/35389

Beck, S. C. (1971). *A Description of the Voltage to Position Converter*. Retrieved from http://www.vasulka.org/archive/Artists1/Beck,Stephen/VoltToPosition.pdf

Candy, L. (2006). Practice Based Research: A Guide. *CCS Report*, *V1.0*. Retrieved from http://www.creativityandcognition.com/wp-content/uploads/2011/04/PBR-Guide-1.1-2006.pdf

Carvalho, A. (2015). Live Audiovisual Performance and Documentation. In V. Crisp & G. M. Gonring (Eds.), *Besides the Screen: Moving Images through Distribution, Promotion and Curation* (pp. 162–176). https://doi.org/10.1057/9781137471024_9

Carvalho, A., & Lund, C. (Eds.). (2015). *The Audiovisual Breakthrough*. Retrieved from http://www.ephemeral-expanded.net/audiovisualbreakthrough/

Caulfield, B. (2018, March 19). What's the Difference Between Ray Tracing, Rasterization? | NVIDIA Blog. Retrieved March 25, 2019, from The Official NVIDIA Blog website: https://blogs.nvidia.com/blog/2018/03/19/whats-difference-between-ray-tracing-rasterization/

CD405xB Datasheet. (2017). Retrieved March 19, 2019, from http://www.ti.com/lit/ds/symlink/cd4051b.pdf

Chion, M. (1994). *Audio-Vision: Sound on Screen*. New York: Columbia University Press.

Chowning, J. M. (1973). The Synthesis of Complex Audio Spectra by Means of Frequency Modulation. *Journal of the Audio Engineering Society*, *21*(7), 526–534.

Ciufo, T. (2002). Real-Time Sound/Image Manipulation and Mapping in a Performance Setting. *Proc. MAXIS Festival of Sound and Experimental Music*.

Collopy, P. S. (2014). Video Synthesizers: From Analog Computing to Digital Art. *IEEE Annals of the History of Computing*, *36*(4), 74–86. https://doi.org/10.1109/MAHC.2014.62

Connor, N. O. (n.d.). *Reconnections: Electroacoustic Music & Modular Synthesis Revival*.

Dalgleish, M. (n.d.). CEC — eContact! 17.4 — The Modular Synthesizer Divided: The keyboard and its discontents by Mat Dalgleish. Retrieved March 7, 2019, from CEC | Canadian Electroacoustic Community website: https://www.econtact.ca/17_4/dalgleish_keyboard.html

Dunn, D. (1992). *Eigenwelt Der Apparate-Welt Pioneers of Electronic Art*. The Vasulkas.

ETC. (n.d.). Retrieved March 7, 2019, from https://www.critterandguitari.com/etc

Fantinatto, R. (2013). I dream of wires. *DVD. Jason Amm (Producer)*.

Farnell, A. (2010). *Designing Sound*. MIT Press.

Fasciani, S., & Rahman, H. (2018). Tangible Virtual Patch Cords. *University of Wollongong in Dubai - Papers*, 316–321.

Franco, E., Griffith, N. J. L., & Fernström, M. (2004a). *Issues for Designing a flexible expressive audiovisual system for real-time performance & composition*.

Franco, E., Griffith, N. J. L., & Fernström, M. (2004b). Issues for Designing a flexible expressive audiovisual system for real-time performance & composition. *NIME*.

gillet, émilie. (2019). *Eurorack modules. Contribute to pichenettes/eurorack development by creating an account on GitHub* [C++]. Retrieved from https://github.com/pichenettes/eurorack (Original work published 2013)

GLES/Raspberry Pi shaders on desktop? - arm - openFrameworks. (n.d.). Retrieved February 9, 2019, from https://forum.openframeworks.cc/t/gles-raspberry-pi-shaders-on-desktop/28837

Global Music Production Software Market to Triple in Value in the Next Five Years. (2018, July 6). Retrieved March 27, 2019, from audioXpress website: https://www.audioxpress.com/news/global-music-production-software-market-to-triple-in-value-in-the-next-five-years

Grierson, M. (2005). *Audiovisual Composition* (University of Kent). Retrieved from http://www.strangeloop.co.uk/Dr.%20M.Grierson%20-%20Audiovisual%20Composition%20Thesis.pdf

Holzer, D. (2017). macumbista.net » Vector Synthesis. Retrieved March 27, 2019, from http://macumbista.net/?page_id=4869

Ken Stone's Modular Synthesizer. (n.d.). Retrieved March 28, 2019, from http://www.elby-designs.com/webtek/cgs/cgsld/cgsld.html

Levin, G. (2000). *Painterly Interfaces for Audiovisual Performance* (M.Sc. Dissertation. Massachusetts Institute of Technology). Retrieved from http://acg.media.mit.edu/people/golan/thesis/thesis300.pdf

Levin, G. (2009). Audiovisual Software Art: A Partial History. *Recuperado a Partir de Http://Www. Flong. Com/Texts/Essays/See_this_sound_old*.

Longobardi, K. (2015, 2018). kashimAstro/ofxGPIO. Retrieved March 14, 2019, from https://github.com/kashimAstro/ofxGPIO

LZX Industries Product Documentation. (2019). Retrieved from https://github.com/lzxindustries/documentation (Original work published 2018)

Magnusson, T. (2010). Designing Constraints: Composing and Performing with Digital Musical Systems. *Computer Music Journal*, *34*(4), 62–73. https://doi.org/10.1162/COMJ_a_00026

Manovich, L. (2018). How Media Became New. In P. Heyer & P. Urquhart (Eds.), *Communication in History* (7th ed., pp. 293–296; By D. Crowley, P. Urquhart, & P. Heyer). https://doi.org/10.4324/9781315189840-42

MCP3004/3008 Datasheet. (2008). Retrieved March 19, 2019, from https://cdn-shop.adafruit.com/datasheets/MCP3008.pdf

Moritz, W. (1997, April). The Dream of Color Music, And Machines That Made it Possible. *Animation World Magazine*, (2.1). Retrieved from http://www.awn.com/mag/issue2.1/articles/moritz2.1.html

Op-amp Comparator and the Op-amp Comparator Circuit. (2015, May 25). Retrieved March 19, 2019, from Basic Electronics Tutorials website: https://www.electronics-tutorials.ws/opamp/op-amp-comparator.html

Patricio, V. G. (n.d.). The Book of Shaders. Retrieved February 8, 2019, from The Book of Shaders website: https://thebookofshaders.com/01/

Puckette, M. (2002). Max at seventeen. *Computer Music Journal*, *26*(4), 31–43.

Ribas, L. (2011). *The Nature of Sound-Image Relations in Digital Interactive Systems* (PhD thesis). Universidade do Porto Faculdade de Belas Artes, Porto.

Rijnieks, K. (2015). *OfxPiMapper—Projection mapping tool for the Raspberry Pi*. Retrieved from https://aaltodoc.aalto.fi:443/handle/123456789/18589

Savov, V. (2017, June 30). Unity is the little game engine that could revolutionize animated movies. Retrieved March 25, 2019, from The Verge website: https://www.theverge.com/2017/6/30/15899446/unity-cinemachine-unite-europe-2017-animation

Sen, S. (2018). sourya-sen/Aalto_Day_One: Code repo for Aalto Day One branding elements. Retrieved March 7, 2019, from https://github.com/sourya-sen/Aalto_Day_One

Sen/Oblique, S. (2015). *Berlin: Not At War // Audio Visual EP*. Retrieved from https://vimeo.com/129318180

Skains, R. L. (2018). Creative Practice as Research: Discourse on Methodology. *Media Practice and Education*, *19*(1), 82–97. https://doi.org/10.1080/14682753.2017.1362175

Sound and Physical Interaction | SOPI Research Group. (n.d.). Retrieved March 27, 2019, from SOPI Research Group website: https://sopi.aalto.fi/

SOUNDEST Zine. (n.d.). Retrieved March 17, 2019, from AVA GRAYSON website: http://www.aigrayson.com/soundest

Sutherland, I. E. (1964). Sketchpad a Man-Machine Graphical Communication System. *SIMULATION*, *2*(5), R-3. https://doi.org/10.1177/003754976400200514

Technical Details A-100. (n.d.). Retrieved March 14, 2019, from http://www.doepfer.de/a100_man/a100t_e.htm

Three Methods to Filter Noisy Arduino Measurements. (2017, May 20). Retrieved February 9, 2019, from MegunoLink website: https://www.megunolink.com/articles/coding/3-methods-filter-noisy-arduino-measurements/

Trocco, F., & Pinch, T. J. (2009). *Analog Days: The Invention and Impact of the Moog Synthesizer*. Harvard University Press.

Unity Technologies. (n.d.). Real-Time Filmmaking, Explained. Retrieved March 25, 2019, from Unity website: https://unity.com/solutions/film/real-time-filmmaking-explained

Whalley, B. (2009). *Synth Britannia*. Retrieved from http://www.imdb.com/title/tt1658487/

Whitney, J. (1980). *Digital Harmony: On the Complementarity of Music and Visual Art*. Retrieved from http://archive.org/details/DigitalHarmony_201611

Whitwell, T. (2018). *TomWhitwell/MTM-Parts-Library on Github*. Retrieved from https://github.com/TomWhitwell/MTM-Parts-Library (Original work published 2015)

Whitwell, T. (2019). *Virtual Radio module for Eurorack . Contribute to TomWhitwell/RadioMusic development by creating an account on GitHub* [Eagle]. Retrieved from https://github.com/TomWhitwell/RadioMusic (Original work published 2014)

Zicarelli, D. (1991). Communicating with Meaningless Numbers. *Computer Music Journal*, *15*(4), 74–77. https://doi.org/10.2307/3681077